

# ConcreteBending 8.0

## User's Guide



---

Updated: 12/10/2024

Copyright © 1994-2024 IES, Inc. All rights reserved.

## Table of Contents

1.	Introduction	3
1.1.	Welcome to ConcreteBending 8.0	3
1.2.	Features	4-5
1.3.	Program Layout	5-8
1.4.	Upgrade Guide	9
1.5.	Release History	10-14
1.6.	Preferences	14-15
1.7.	Support Resources	15
2.	Modeling	16
2.1.	Slabs and Walls	16-20
2.2.	Point Supports	20-21
2.3.	Line Supports	21-22
2.4.	Beams	22-26
2.5.	Load Combination Criteria	26
2.6.	Loads	26-27
2.7.	Self Weight	27
2.8.	Load Points	27-29
2.9.	Analysis	29-31
2.10.	Punching Shear	31-37
3.	Report	38
3.1.	Reports	38
3.2.	Tables	38-39
3.3.	Saved Reports	39-40
3.4.	Beam Graphs	40-41
4.	Integration	42
4.1.	IES VisualAnalysis	42
4.2.	IES VAConnect	42

5. Script	43
5.1. Script Overview	43-48
5.2. Commands	48-69
5.3. External Scripts	69
5.4. Example: Two-Way Slab Generator	69-70
5.5. Example: Refine Mesh	70-71
5.6. Example: Optimize Thickness	71-72

# ConcreteBending 8.0 User's Guide

## 1 Introduction

### 1.1 Welcome to ConcreteBending 8.0

ConcreteBending will help you analyze and design a wide variety of concrete elements in flexure. The design of elevated slabs, beams, tank walls, foundation walls, and many other common concrete structures is made more efficient in ConcreteBending. It automates much of the work involved in finite element modeling that can be tedious in a more general tool. The software will analyze your concrete model to determine moments, shears, and displacements using finite element analysis, and then help you ensure your structure meets the design requirements of ACI 318, ACI 350, or CSA A23.3.

#### Getting Started

- Use **File | Open** Example to see sample projects.
- [Feature List](#)
- [Program Layout](#)
- [Upgrade Guide \(what's new\)](#)
- [FAQ Answers](#) at iesweb.com for business, licensing, installation issues.

#### Help Notation

- Menu items appear like this: **File | New**.
- Keystrokes or mouse commands appear like this: **Shift+Click**.

#### Disclaimer

ConcreteBending is a proprietary computer program of Integrated Engineering Software (IES, Inc.) of Bozeman, MT. This product is intended for use by licensed, practicing engineers who are educated in structural engineering, students in this field, and related professionals (e.g. Architects, Building Inspectors, Mechanical Engineers, etc.). Although every effort has been made to ensure the accuracy of this program and its documentation, IES, Inc. does not accept responsibility for any mistake, error, or misrepresentation in, or as a result of, the usage of this program and its documentation. (Though we will make every effort to ensure that problems that we can correct are dealt with promptly.) The results obtained from the use of this program should not be substituted for sound engineering judgment.

#### License and Copy Restrictions

By installing ConcreteBending on your computer, you become a registered user of the software. The ConcreteBending program is the copyrighted property of IES, Inc. and is provided for the exclusive use of each licensee. You may copy the program for backup purposes and you may install it on any computer allowed in the license agreement. Distributing the program to coworkers, friends, or duplicating it for other distribution violates the copyright laws of the United States. Future enhancements and technical support depend on your cooperation in this regard. Additional licenses and/or copies of ConcreteBending may be purchased directly from IES, Inc.

#### IES, Inc.

Integrated Engineering Software, Inc.  
3740 Equestrian Ln Ste 1  
Bozeman, MT 59718

**Sales or Licensing:** 406-586-8988, [sales@iesweb.com](mailto:sales@iesweb.com)

**Technical Support:** [support@iesweb.com](mailto:support@iesweb.com)

## 1.2 Features

### General

- Simple, standard Windows interface for easy navigation
- Unlimited Undo & Redo commands
- Work in any unit system, perform math on input, use custom unit 'styles'
- Program is self-documenting with tooltips on commands and input parameters
- Numerous preference settings for better defaults
- Drive the program with the Command Line
- Run External Scripts to automate common tasks and more
- Free training videos provided for learning efficiency
- Free technical support email with fast, friendly turnaround

### Modeling

- Quick-start with typical geometries
- Create any slab geometry by sketching boundaries
- Import from CAD for complex and detailed boundaries
- Work with column-lines to organize and modify structures
- Create slabs or walls with various thicknesses and reinforcing settings
- Define concrete beams
- Define point supports (e.g. columns)
- Define line supports (e.g. walls)
- Easily refine finite element mesh
- Create new model objects by copying existing items

### Loading

- Model objects can be loaded in multiple service load cases (e.g. Dead, Live, etc.)
- Automatic building code load combinations are available
- Includes IBC, ASCE 7, and NBC Load Combinations
- Customizable building code combinations (see Load Case Manager)
- Create custom load combinations in any project
- Apply point loads, line/distributed loads, and area load pressures to the structure
- Copy and paste loads to objects
- Copy and scale loads to other load cases

### Analysis

- FEA model is constructed automatically by the software
- Automated "background" analysis is fast
- Plate elements use a "thick-plate" formulation for accurate shear results
- Advanced error-checking and reporting
- Export project to VisualAnalysis to better understand model details or for more sophisticated analysis

# ConcreteBending 8.0 User's Guide

## Design

- Implements ACI 318-19, ACI 318-14, CSA A23.3:19, CSA A23.3-14, ACI 350-20, and ACI 350-06 specifications
- Concrete boundary checks for bending, minimum steel, one-way shear, and punching shear
- Environmental (water holding) structure design checks
- Specify reinforcing parameters for each concrete boundary in the model
- Specify a reinforcing pattern for the program to check or have ConcreteBending select an optimal pattern
- Detailed concrete beam design checks
- Export loads to VACONnect (sold separately) for base plate design

## Reporting

- Quick Full Report includes graphics and all details, with options
- Custom reporting to include just the information you need
- Print Preview mode while working with reports
- Paste any graphics into your report
- Customizable page margins, fonts, colors
- Use your own company logo in report page headers
- Print to any printer including PDF
- Export to text clipboard or save to other formats like .xlsx

## Limitations

- Does not consider reinforcing development lengths, cutoffs, bends, etc.
- Does not produce structural drawings
- Cannot model a system of disconnected slabs.

## Be a Squeaky Wheel

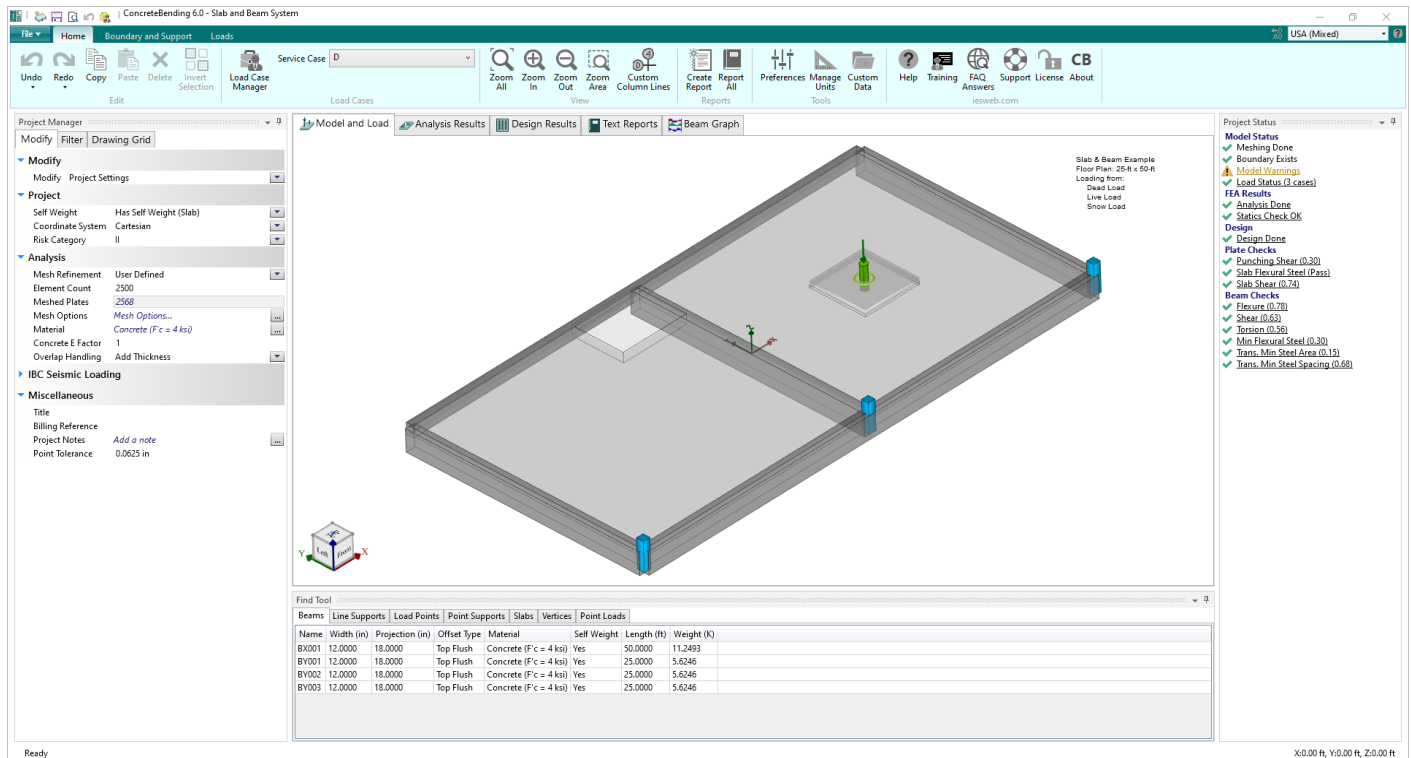
If you need a new feature, please let us know! We are always looking for ways to improve products in ways that you desire. See [Technical Support](#).

## 1.3 Program Layout

The best way to learn ConcreteBending is to use and explore the program to get to know what is available under each button or menu. Several [Tutorial Videos](#) are also available which explain many features of the software.

### Screen Layout

The image below introduces the program terminology used in this help file and the training videos. Panels may be resized by dragging their dividers or repositioned by dragging their title bars or right-clicking on the title. Use the "pushpin" icon to collapse panels temporarily to gain more space for working. Hold the mouse pointer over the screen image below for information about each area of the program.



## Title Bar

The title bar displays the version of ConcreteBending and the file name of the project. Also, there are helpful buttons in addition to the Windows system buttons.

## Main Menu / Toolbar

The main menu, Toolbar, or Ribbon, contains various commands to direct ConcreteBending. Each is organized within a group to help locate them quickly. Each command has a description which appears when the mouse pointer is hovered over it. Many have hot-key shortcuts.

## Project Manager

The Project Manager provides immediate access to frequent operations in ConcreteBending. This tool is docked on the left side of the window by default and displays various tabs depending on the active window. This window can be docked on the left or made to float independently if more space is needed to work. Alternatively, drag the side border to make it wider or narrower.

- The **Modify** tabs are used to change the project settings or the properties of selected objects in the Model and Load View or the Design Results View.
- The **Filter** tab is used to control what is shown or hidden in the active view.
- The **Drawing Grid** tab is used to control the Sketch Grid to aid in drawing models in the Model and Load View.
- The **Result** tab replaces the Modify tab when the Analysis Result View is active. This tab provides key result information for the active load case.
- The **Tables** tab is used to add available tables to the report.
- The **Report Filter** tab is used to define the report settings, apply the model filters, modify the parameters of the table selected in the report, and select which table columns to display.

# ConcreteBending 8.0 User's Guide

## Graphic Views

These views provide a way to view the model, analysis results, design results, and reports. Each tab displays different options and will provide different information in the Project Manager and Find Tool. Some Graphic tabs will only appear based on objects in your model, such as the Beam diagrams.

## Project Status

This panel provides a quick update on what is done, what is in-progress, and whether things are working or failing in your model or checks. Click on any item that is underlined for more information, a report, or a dialog containing quick actions.

## Pipeline Status

Shows background meshing/analysis/design progress. Background processing is done on a separate thread of your processor so you may continue working while they run. The only time you need to wait for the program is when the mouse cursor changes into an hour-glass or if you wish to view the analysis or design checks that are currently in-progress. Detailed progress bars are available for background activity by clicking on the status-bar at the bottom of the screen.

## Find Tool

The Find Tool provides an efficient way to view, select, and edit slabs, members, supports, loads, etc. This tool is docked on the bottom of the window by default. Use **F7** or the push-pin icon to auto-hide this panel. When docked, drag the side border to make the panel larger or smaller. The Find tool allows you to find, select, edit, and delete objects even if they are not visible in the active window. **Double-click** on an element (slab, pile, pier, etc) and the graphics window will zoom-in to show that element, if it is visible. Lists shown in the Find tool can be sorted by clicking on a column header (**click** again to reverse the order). Select items just like any list in Windows using the **Shift** and **Ctrl** keys to select a range or to toggle individual items.

## Units & Precision

Above the toolbar on the far right is the Units drop-down for selecting the way physical quantities are displayed. Change the number of decimal places or significant digits using the icon to the left of the unit selector. Go to **Home | Manage Units** to create custom unit styles or edit existing unit styles.

## Data Entry: Physical Quantities

Enter values in any unit style. Enter any number or math expressions followed by a known abbreviation. Length units may be entered in "ft-in-16ths" notation as well. Entered values are converted and then redisplayed in the current 'display' units.

## Mouse and Keyboard Commands

### Selection:

- **Click** to select (mouse hover indicates what object will be selected)
- **Click** in the 'whitespace' of a view to unselect everything and access Project Settings
- **Ctrl+Click** to toggle object selection without affecting other objects
- **Shift+Click** to select all objects of a given type
- **Shift+Drag** draw a selection box (left-to right selects fully enclosed objects, right-to-left selects any partially enclosed objects)
- **Shift+Ctrl+Click** to select items of one type with the same Name Prefix as the item clicked on



## Zoom:

- **Scroll Mouse Wheel** with the pointer over the point to zoom in or out
- **Double Click Mouse Wheel** to Zoom All
- **Ctrl+** (plus) and **Ctrl-** (minus) keys
- **Ctrl+Home** for zoom all/extents
- **Ctrl+End** to enable the **Home | Zoom Area** command then **Drag** to create the Area

## Pan:

- **Drag Mouse Wheel** to pan
- **Shift+Arrow** keys will also pan

## Rotate:

- **Ctrl+Drag Mouse Wheel** to rotate the view
- **Click** on a face, edge or corner of the Cube in the lower-left corner of the graphics to rotate the view
- **Ctrl+Arrow** keys will also rotate

## Context Menu:

- **Right-Click** the mouse for a short menu of relevant commands based on the view and what is selected
- **Shift+F10** also display the context menu

## Hot Keys:

- **Alt** will expose the hot-keys in the main menu
- **F1** Help
- **F7** Show or hide the Find Tool
- **F9** Toggle Picture View
- **Esc** Cancel the Graphic drawing and enter the Draw Nothing mode
- **Delete** the Graphic selection
- **Ctrl+C** Copy graphic image to clipboard
- **Ctrl+V** Generate copies, or paste graphics in Report View

## Miscellaneous:

- **Drag** in the Model View to sketch slabs, grade beams, walls, etc.
- **Double Clicking** in the Analysis Result will generate a Text Report for the object. Double-clicking on an element or node in the Find Tool will Zoom to that item.

## Context Menu:

**Right-Click** the mouse for a short menu of relevant commands based on the view and what is selected.

## Middle-Mouse "Button" in Windows

Depending on your system, you may need to go into Control Panel, Hardware, Mouse, and set the wheel button to behave like a "middle button click". Some mouse utility programs may override that setting or it may not be set up on some versions of Windows.

## 1.4 Upgrade Guide

### Version 8.0 (January 2025)

#### Punching Shear Design Overhaul

- Enhanced punching groups
  - Option added to check punching subgroups
  - Line Supports can now be included in punching shear groups
  - Smarter punching shear group generation
  - Smarter punching group/grade beam interaction
  - More accurate critical sections for punching shear
- Improved how "d" is calculated
  - Accounts for the direction of punching
  - Accounts rebar sizes and orientations
  - Use correct value based on the specified or optimized design approach
- Critical section now minimizes the punching perimeter ( $b_o$ )
- More accurate punching shear demand calculation
- All components of the punching free body diagram can now be reported
- Punching Shear design moved to background thread (UI stays responsive while design works)
- Improved graphics performance for punching boundaries
- More accurate beta calculation

#### New Design Specification Options

- Added option to specify  $\gamma$  for ACI 350-20 & ACI 350-06
- Added option to neglect the size effect factor (i.e.  $\lambda_s=1$ ) for ACI 318-19

#### New Load Combinations

- IBC 2024 Load Combinations
- NBC 2020 Load Combinations

#### New Scripting Commands

- Set project criteria (title, billing, notes, etc.)
- Delete loads
- Generate custom load combinations
- Extract results on a per boundary basis
- Set visible result case
- Add graphics to report
- Obtain lists of the service and strength result cases

#### New Report Features

- Added option to save reports in the project
- Added option to save reports as a style
- Reorganized report project manager

## 1.5 Release History

### Overview

#### Versions

- Version 8.0 released January 2025
- Version 7.0 released September 2023
- Version 6.0 released November 2021
- Version 5.0 released November 2018
- Version 4.0 released October 2017 - First version, built on VisualPlate 3.0 features

### Version 8

#### Punching Shear Design Overhaul

- [Enhanced punching groups](#)
  - Option added to check punching subgroups
  - Line Supports can now be included in punching shear groups
  - Smarter punching shear group generation
  - Smarter punching group/grade beam interaction
  - More accurate critical sections for punching shear
- [Improved how "d" is calculated](#)
  - Accounts for the direction of punching
  - Accounts rebar sizes and orientations
  - Use correct value based on the specified or optimized design approach
- Critical section now minimizes the punching perimeter ( $b_o$ )
- More accurate punching shear demand calculation
- All components of the punching free body diagram can now be reported
- Punching Shear design moved to background thread (UI stays responsive while design works)
- Improved graphics performance for punching boundaries
- More accurate beta calculation

#### New Design Specification Options

- Added option to specify  $\gamma$  for ACI 350-20 & ACI 350-06
- Added option to neglect the size effect factor (i.e.  $\lambda_s=1$ ) for ACI 318-19

#### New Load Combinations

- IBC 2024 Load Combinations
- NBC 2020 Load Combinations

#### New Scripting Commands

- Set project criteria (title, billing, notes, etc.)
- Delete loads
- Generate custom load combinations
- Extract results on a per boundary basis
- Set visible result case
- Add graphics to report
- Obtain lists of the service and strength result cases

## New Report Features

- Added option to save reports in the project
- Added option to save reports as a style
- Reorganized report project manager

## Version 7

### Scripts & Command Line

- [Command Line](#) is a powerful new way to drive the program:
  - Definite slabs, beams, point supports, line supports etc.
  - Apply loads to regions, load points, and beams
  - Adjust the analysis settings
  - Extract plate results (displacements, shears, moments, etc.)
  - Add tables to report and export reports
- [External Script](#) files can automate common tasks and much more:
  - Generate boundaries, apply loads, extract results
  - Refine the mesh until model converges
  - Perform model optimization

### New Design Specifications

- ACI 318-19
- ACI 350-20
- CSA A23.3:19

### Other Features

- Plate design diagrams
- Query plate results at a specified location
- Beam loads remain constant when switching between resultant to distributed loads
- Table of Contents report table
- Loaded Area information added to Area Uniform Loads tab in the Find Tool
- Updated Microsoft .NET framework for features and performance

## Version 6

## Key Features

- Slab's mesh can be refined at load point and point support locations
- Punching shear reinforcement can be specified

## Other Features

- Improved Design Rebar Patterns dialog that retains rebar patterns
- Plate result diagram overhaul
- Program ensures area loads are properly modeled before starting the analysis
- The number of shear legs can be specified for beams
- Improved filtering of table extremes
- Justification can be specified for report tables
- Improved drawing grids
- Improved the performance of generated loads

## Fixes & Minor Changes

- Improved Help File documentation
- Crash recovery file improvements
- Fixed concrete preferences
- Fixed point support size reporting error

## Version 5

### General

- Graphic wire frame in picture view mode
- Project Manager: Categories remember last open/collapsed state
- Project Manager: drop-lists are activated by clicking anywhere, not just on arrow
- Improved warning message formatting
- Improved Print Preview display for graphics
- Column-line graphics are "filtered" to avoid clutter
- Improved area load graphics with shaded tops
- Graphics performance is **up to 20x faster**
- Removed 'memory leaks', which slowed program over time

### Modeling

- Ability to override punching perimeter
- Ability to insert a vertex along an area side
- DXF import shows bounds and an option for centering at the origin
- Polygon boundary defined by side length in addition to radius

### Loading

- Ability to move area loads by side coordinates

# ConcreteBending 8.0 User's Guide

- Beam loads can now be applied relative to wall orientation (parallel and perpendicular)
- Multiple selection in Load Case Manager

## Analysis

- Line supports now insert a "drilling" spring support
- Individual service cases can be analyzed
- Analysis is now **12% faster**

## Design

- **Canadian CSA A23.3-14** specification for slabs and beams
- **ACI 350-06** Environmental Engineering Concrete Structures for slabs
- **Slab reinforcement may be specified** (rather than patterns to search/check)
- **Distinct slab design parameters for each slab-boundary**
- Filter beam design results graphically by limit state
- Plate design checks are **1.5x faster**

## Reporting

- **Name filters** are now enabled for both graphic and report filters
- Unavailable tables shown disabled with reason for not being available
- Table drop position used to locate when adding new tables in Text Reports
- Full report can now have categories filtered (model, loads, results, design, graphics)
- Flexure and Shear Design Report Tables now include key intermediate design details

## Version 4

Built on the concrete design module of VisualPlate 3.0

### General

- 64-bit implementation, no more out-of-memory issues
- Multiple-threaded architecture uses all processor cores
- New UI, Ribbon toolbar, consistent with other IES tools
- [3D model/viewing](#) (ctrl+mouse wheel or click the cube)
- [Preference](#) settings (fonts, colors, sizes, options, etc.)
- History Flies (automatic daily backups of a project-file, see preferences)
- Your Logo in a Text Report (see preferences)

### Modeling

- No merge/intersection necessary for concrete boundaries
- Easier to edit
- Automatic split/connect meshing for crossing or overlapping line supports, beams
- Control over overlapping concrete thicknesses
- DXF import has insertion point so huge coordinate files can be moved.

### Loading

- Implements ASCE 7 load combinations
- More Direct full-boundary loading

## Analysis

- Automated behind-the-scenes
- Much faster analysis (using all processor cores)
- Accurate results

## Design

- Implements ACI 318-14 specification
- Beam design, limit states and unity check
- Concrete boundary reinforcing design
- Custom detailing regions
- One-way and two-way punching shear calculations

## Reporting

- Powerful Report Viewer
- Many tables and options
- Saved reports in project files
- Paste graphics into reports
- Complete project report
- Predefined, customizable reports

## 1.6 Preferences

ConcreteBending preferences are default settings that primarily affect the behavior of new projects. These are not project-specific settings, which are found in the [Project Manager](#). The preference settings can be adjusted through **Home | Preferences**. Some settings do not take effect until a new project is created or until the program is restarted. Use the Restore All Defaults button to restore the ConcreteBending preference settings to their original state. While most of the preference settings are self-explanatory, a few are documented below. Preference settings are saved on your machine in the IES folder: `C:\Users\<your.login>\AppData\Local\IES\Customer`.

### Project, History Files

ConcreteBending can create up to 10 dated-backup copies of a project-file. These are stored in the IES\Customer\HistoryProjects folder. If you have data corruption or lost a project, you might be able to recover from one of these. You can turn this feature off (to save disk space or use your own backup mechanisms) by setting the value to zero.

### Project, Next Inspector Field on Enter

By default, in the Project Manager, Modify area, when you press the Enter key it sets your input data but does not change fields. The Tab key lets you move from one field to another. You can enable the movement to the next field on the Enter key with this option.

### Reports, Maximum Page Count

# ConcreteBending 8.0 User's Guide

Reports can become quite large and create performance issues. ConcreteBending allows you to set the maximum number of pages allowed in a report before truncation occurs and a warning appears.

## 1.7 Support Resources

### Did you Search this Help File?

Take advantage of the help and support built into the software, as described in the [Program Layout](#) section of the User's Guide. This document can be searched, and you should try different potential terms, sometimes less is more when searching (use just the unique word or words). A Table of Contents is also available.

### Do Not Contact Support For:

- **Licensing/Sales.** Use [www.iesweb.com](http://www.iesweb.com) or [sales@iesweb.com](mailto:sales@iesweb.com).
- **Modeling Advice.** Determining how to model a structure is your responsibility as an engineer.
- **Model Validation.** IES cannot validate your model or your results. If you can document a software defect, contact support and we will investigate further and create fixes as necessary.
- **Engineering Theory.** IES is not in the business of educating engineers. There are textbooks referenced in this help file.

### Technical Support

- **Support Email:** [support@iesweb.com](mailto:support@iesweb.com). Replies are usually within 2 business hours, if you don't hear anything within a business day, assume it got spam filtered or lost and follow-up. For best results, be sure to ask a question, indicate exactly which IES product & version you are using, include as much detail as is practical. If relevant, please attach a project file and/or screenshots.
- **Support Telephone:** Not Available. We have found this to be too inefficient for everybody. With email you can attach a screen shot, a project file, and we can better direct your question to the IES expert for that product or area. Phone tag takes longer than you think.
- **Business Questions:** For any licensing or sales-related questions or issues contact [sales@iesweb.com](mailto:sales@iesweb.com).



## 2 Modeling

### 2.1 Slabs and Walls

Concrete boundaries (also known as slabs or walls) are the primary elements modeled and designed in ConcreteBending. Boundaries with a wide variety of shapes and sizes can be modeled in the program. The concrete boundary can be supported by [Point Supports](#) and [Line Support](#) and stiffness can be added by modeling [Beams](#) into the boundary. A single concrete material is used for all boundaries in the project and is defined in the [Project Settings](#).

#### Modeling

ConcreteBending is used to model concrete boundaries that bend out-of-plane which generally fall into two categories: slabs and walls. A slab's self weight will cause it to bend out of plane whereas a wall's self weight will not. The project can be set to include the self weight (for slab type systems) or exclude the self weight (for wall type systems) in the Project Settings (located in the [Project Manager | Modify](#) tab when nothing is selected in the Model and Load view). In this section the concrete boundaries are often referred to as "Slabs", however, the term "Wall" is interchangeable.

#### Slabs/Walls

In the Model and Load view, circular, rectangular, polygonal, or custom concrete boundaries can be drawn using the buttons in the ribbon. These slabs are defined by the vertices at boundary points. New concrete slabs can be drawn on grids or existing vertices and snap points can be used to create the boundary. The number of snap points along each edge of the slab is specified in the [Project Manager | Filter](#) tab. Any arbitrary geometry can be constructed by drawing multiple slabs connected to or overlapping each other. Adjoining slabs are modeled as continuous, with common displacements and flexural rotations at the boundaries. Each individual slab boundary can have a different thickness. At locations of overlapping slabs, the thickness is specified using the Thickness Overlap parameter in the [Project Manager | Modify](#) tab. Note: Disconnected slabs are not allowed in ConcreteBending and expansion-joints or other discontinuities in the slab cannot be modeled.

#### Holes

A selected slab can be turned into a hole by setting Hole? = Yes in the [Project Manager | Modify](#) tab. Loads that exist within holes are not included in the analysis. If portions of Area Loads lie partially over holes, only the loading lying over slabs are considered (i.e. the loading that occurs over holes is not distributed to adjacent plate elements).

#### Reinforcement Details

The reinforcement parameters for each slab are defined by selecting the slab from the Model and Load view and editing the *Reinforcement Details* section in the [Project Manager | Modify](#) tab, as described below.

- **Bar Configuration:** Double Mat or Single Mat. Determines if X & Y reinforcement is placed in a single or double layer in the concrete slab.
  - *Double Mat Parameters*
    - **Top / Bottom Layer Outer Bars:** The direction, X or Y, of the outer reinforcing bars. Outer bars are the bars closest to the top/bottom face of the slab.
    - **Top / Bottom Cover:** The clear cover from the top/bottom face of the slab to the outer reinforcing bars.
  - *Single Mat Parameters*
    - **Top Bars:** The direction of the Top reinforcement bars. The Top bars are the bars closest to the Global +Z face of the slab.
    - **Bottom Cover:** The clear cover from the bottom face of the slab to the reinforcing bars.

# ConcreteBending 8.0 User's Guide

- **Design Approach:** Optimize or Specify. The Optimize design approach allows ConcreteBending to search for an optimal reinforcement pattern that meets all design requirements from a pre-defined list of desired patterns. The Specify design approach allows one reinforcement pattern for each direction and layer to be specified explicitly and checked during the design process. Further information of how this affects the design process can be found below in the discussion on Design Flexure Checks (below).
  - *Optimize Approach*
    - **Rebar Patterns:** Defines the desired rebar patterns available for the slab during the design process.
  - *Specify Approach*
    - **Size:** The size of the reinforcement in the X & Y directions and for the Top and Bottom, or Single layers, depending on the Bar Configuration selected.
    - **Spacing:** The spacing of the reinforcement in the X & Y directions and for the Top and Bottom, or Single layers, depending on the Bar Configuration selected.

## Loading

Slabs may be loaded with Full Area, Circular, Rectangular, Tubular, and Ring area loads or by concentrated Point loads. Loads can also be transferred to the slab by [Beams](#).

## Analysis

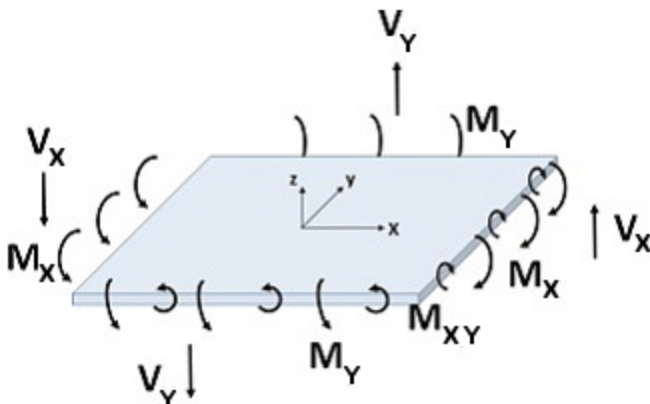
[Finite element analysis](#) is approximate and numerical. Use [mesh refinement](#) and examine results carefully, before trusting design checks.

## FEA Model

Slabs in ConcreteBending are meshed into triangular FEA plate elements. See the [Analysis](#) page for more information on the FEA mesh density, including the available settings and options. For a better understanding of the FEA model that is built by ConcreteBending, use the [File | Export a VA Project](#) feature to create a VisualAnalysis project and examine the FEA model in more detail. More information on integration between ConcreteBending and both [VisualAnalysis](#) and [VAConnect](#) can be found in the Integration section of the Help File.

## Plate Results Sign Convention

1. Positive moments put the slab's top bars (Global +Z) in tension.
2.  $M_X$  and  $V_X$  moments and shears act on the Global X-face of the plate elements.
3.  $M_Y$  and  $V_Y$  moments and shears act on the Global Y-face of the plate elements.
4. Moments are carried by reinforcement that runs parallel to the specified directions, X & Y.



## Viewing Analysis Results

The results from the finite element analysis can be displayed graphically in the Analysis Results view. The **Project Manager | Results** tab displays the numerical results that correspond to the colored graphics. With nothing selected, the results displayed in the **Project Manager** are a summary for the selected result case, whereas if one or more individual plates are selected, the **Project Manager** shows the result range for the selected plates. The various Result Cases that were included in the finite element analysis can be selected using the Result Case drop down from the **Ribbon | Home** tab. The Result Type displayed graphically in the Analysis Result view is specified in the **Project Manager | Result Filter** tab.

## Plate Diagrams

Plate result moment, shear, and displacement diagrams are available in the Analysis Results view. Left-click and drag a line across the slab to view the result at the line's location or right-click and select Show Plate Result Diagram from the context menu to view the diagrams for the selected result case. The slice direction and location can be adjusted within the Plate Result Diagrams dialog box along with the number of plots displayed and the plot type. Note: The result location and plot settings will persist within one session of ConcreteBending. This allows the Plate Result Diagrams dialog box to be closed to select a different result case or to modify the model without losing the adjustments that were made in the dialog.

## Design

ConcreteBending checks and designs concrete slabs and walls according to the following design specifications:

- ACI 318-19
- ACI 318-14
- CSA A23.3:19
- CSA A23.3-14
- ACI 350-20
- ACI 350-06

## Flexure Checks

Slabs that experience two-way action have both bending moments ( $M_X$  and  $M_Y$ ) and twisting moments ( $M_{XY}$ ). The twisting moments may cause the maximum bending demand to not coincide with the X or Y reinforcing directions. To ensure that the slab has adequate strength in all directions, ConcreteBending uses the method by Wood and Armer as explained by MacGregor and Wight<sup>1,2</sup> to design the reinforcement. The following equations are used to calculate the design moments which are obtained from Wood and Armer when  $k=1.0$ . According to MacGregor and Wight,<sup>2</sup>  $k=1.0$  is the best choice for a wide range of moment values. ConcreteBending takes the critical flexure section as the face of Beams, Point Supports, and Line Supports. Plate nodes that fall within these objects are not checked for flexure.

$$M_{X+} = M_X + |M_{XY}| \geq 0$$

$$M_{X-} = M_X - |M_{XY}| \leq 0$$

$$M_{Y+} = M_Y + |M_{XY}| \geq 0$$

$$M_{Y-} = M_Y - |M_{XY}| \leq 0$$

The bending moment capacity of the slab is a function of the compressive strength ( $f'_c$ ), thickness, and the area of steel reinforcement provided. Reinforcing is placed in both plan directions (X & Y) and in either one or two layers. This results in up to four reinforcing quantities at every point in the slab (X, Y, Top, Bottom). ConcreteBending performs moment and steel demand calculations at the finite element model node points. When the slab's reinforcement is set to Optimize (as discussed above), the reinforcement required to meet factored demand is selected by the program based on the list of available rebar patterns. The rebar pattern with the smallest area ( $A_s$ ) is selected from this list that meets the required

# ConcreteBending 8.0 User's Guide

flexural and minimum steel demands. If the slab's reinforcement is set to Specify, the reinforcement pattern defined for the direction and layer in question will be used when performing the design checks.

## One Way Shear Checks

The critical section for one-way shear (i.e. beam shear) is taken at the face of: Load Points, Point Supports, Line Supports, and Beams. If, however, a Point Support or Load Point introduces compression in the slab, the critical section is taken at "d" from the face of the object.

Plate shear checks are reported graphically in the Design Results view by selecting *Shear Unity* from the **Project Manager | Design Information** under the *Slab Details* category (red plate colors indicate a failing unity value). Shear checks can also be reported as numerical results using the programs's Text Reports.

## Punching Shear Checks

Two-way (punching shear) checks are performed at locations where concentrated loads are introduced into the slab, such as at Load Points, Point Supports, and Line Supports. Additional information regarding punching shear design checks and punching shear reinforcement can be found on the [Punching Shear](#) page of the Help File.

## Rebar Design Regions

The software can detail bar patterns by plate element, by column lines or by slab boundary. Use the **Project Manager | Design Filter** tab from the Design Results view to display the different results. Note: The *By Column Line* display is not available if the *Design Approach* is set to *Specify*.

## Reporting Results

Reports can include both text-based and graphical information. Graphical information from the Analysis Results or Design Results views or from the plate diagrams can be inserted into a report using the **Copy** and **Paste** commands.

## Plate Diagrams

Plate result diagrams for moment, shear, and displacement are available in the Analysis Results view and plate design diagrams for shear unity and area of steel required are available in the Design Results view. Left-click and drag a line across the slab to view the diagrams at the line's location or right-click and select Show Plate Result/Design Diagram from the context menu to launch the respective diagram dialog box. The slice direction and location can be adjusted within the dialog along with the number of plots displayed and the plot type. Note: The slice direction and location along with the plot settings will persist within one session of ConcreteBending. This allows the dialog to be closed to select a different result case or to modify the model without losing the adjustments that were made in the dialog. To include graphs in the report, use the copy plots button to copy the plots to the clip board which can then be pasted into the report.

## Text Reports

Analysis and design results can also be viewed in text form using the Report View tab. Once in the Report View, a list of available design tables is shown on the **Project Manager | Tables** tab and can be added to the text report by double-clicking an individual table or dragging and dropping a table into the report. Plates can also be reported on an individual basis by selecting one or more plates from the Analysis Results or Design Results view, and using the right-click context menu to *Report Selected*. Note: The reinforcement settings are included in the *Slabs* table located under the *Structure Tables* category.

## References

1. Wood, Randal H. and Armer, G. S. T., "The Theory of the Strip Method For Design of Slabs," Proceedings, Institution of Civil Engineers, London, Vol. 41, October 1968, pp. 285-313.
2. MacGregor, James G., and Wight, James K., "Reinforced Concrete: Mechanics and Design," Pearson, 2012.

## 2.2 Point Supports

Point Supports in ConcreteBending are used to support the concrete slab at a specified location in the model.

### Modeling

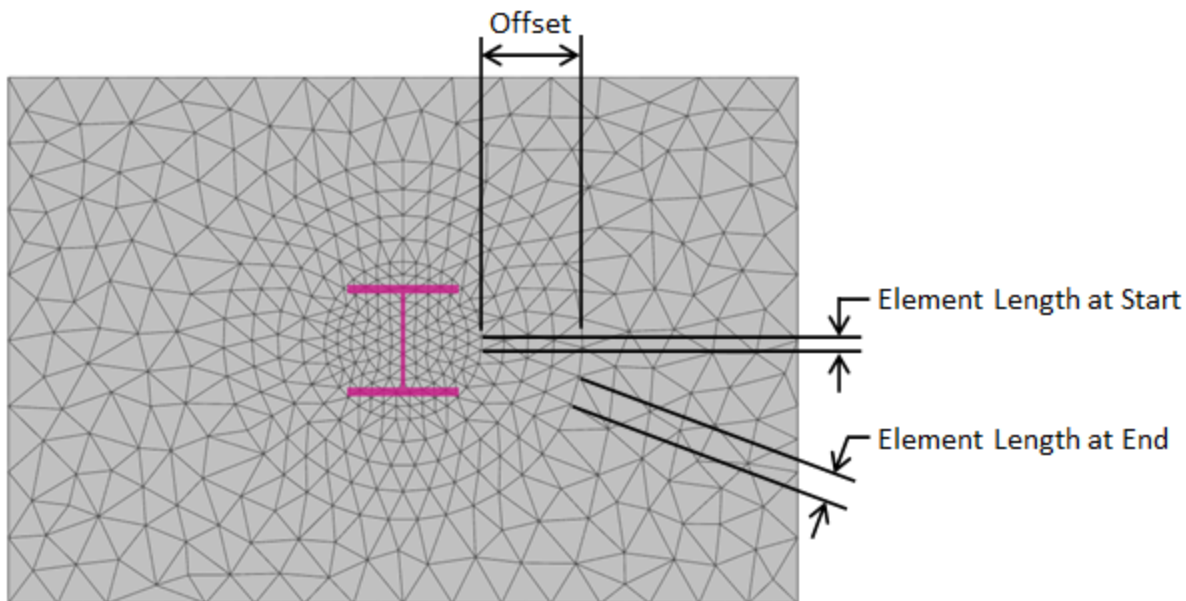
Point Supports are defined by their location, shape, and size in the model. Four shape types are available: Square, Rectangle, Circle, and I Shape. The size and shape of the Point Support can be specified which influences the calculation of the bending and one-way shear checks for a slab supported by the Point Support. Bending in the slab is always calculated at the face of the Point Support. Shear is calculated at the face of the Point Support if the slab is in tension and at a distance "d" away from the face of the Point Support if the slab is in compression. Note: In the Finite Element Analysis (FEA) model, only a single node in the slab's mesh is restrained at a point support; therefore, care should be taken when specifying point supports with large cross-sections.

### Support Type

ConcreteBending only allows deflection in the Z-direction and rotation about the X-axis and Y-axis. Therefore, Point Supports have the option of being set to Fixed, Free, or Specified-K (stiffness) for Force-Z, Moment-X, and Moment-Y.

### Mesh Refinement

The finite element analysis (FEA) method is approximate and the accuracy of the FEA solution typically increases as the mesh becomes finer. Generally, a finer mesh should be used in regions where there are large changes in stresses in the plates (such as at point supports where the support can create stress concentrations). Select a point support(s) and set Refine? = True in the **Project Manager | Modify** tab to refine the mesh in the regions of the chosen point support(s). The mesh refinement parameters for point supports are defined in the image below.



**Point Support Mesh Refinement**

## Loading

Point Supports cannot be directly loaded in ConcreteBending. A load point can be defined at the same location as the point support (i.e. connected to the same node in the finite element model) and load can be applied to the load point.

## Analysis

The force in the Z-direction and the moments in the X-direction and the Y-direction are calculated for each load case and reported for the Point Supports in ConcreteBending. The displacement in the Z-direction and the rotations about the X-axis and the Y-axis are also reported. The slab punching shear results from a Point Support are reported when applicable.

## Design

While point supports appear to be columns when the Picture View is enabled in ConcreteBending, they do not have any length in the finite element model (i.e. they are not modeled as a member element). Therefore, point supports are not designed as axial members in the program. Punching shear design checks, however, are performed for the slab at the location of the point supports.

### Punching Shear

In ConcreteBending, Punching Shear design checks are performed at point support locations and the program can automatically determine the punching shear perimeter (critical section). An individual point support can also define punching shear reinforcement in the form of headed shear studs or shear stirrups located in the slab. Additional information regarding punching shear design checks and punching shear reinforcement can be found in the [Punching Shear](#) section of the Help File.

## Reports

Multiple report tables, which include modeling and result information for the Point Supports, are available in the Text Report view.

## 2.3 Line Supports

Line Supports are used in ConcreteBending to restrain the model along a series of nodes that fall on a straight line which is defined by a start coordinate and an end coordinate in the XY-plane.

### Support Properties

The thickness of the Line Support can be specified and is used with the length of the Line Support to calculate the punching shear perimeter for the two-way shear design check. The bending and one-way shear checks for a slab supported by a Line Supports are calculated at the face of the Line Support. Therefore, adjusting the thickness of the Line Support will change the location where the bending and one-way shear results are reported for the slab. While a Line Support has a defined thickness, it only provides restraint along a single line of nodes in the FEA model. ConcreteBending only allows deflection in the Z-direction and rotation about the X-axis and Y-axis. Line Supports fix the deflection of the nodes on the line in the Z-direction and fix the rotation of the nodes about a line in the XY-plane perpendicular to the Line Support. The three support options (Pinned, Fixed, and Spring) allow the rotational restraint about the line defined by the Line Support in the XY-plane to be specified.

## Analysis Results

The total force in the Z-direction and the total moment about the Line Support is calculated for each load case and

reported in ConcreteBending. The slab punching shear results from a Line Support are reported when applicable.

## Line Support Design

Line Supports are not designed in ConcreteBending.

### Punching Shear

In ConcreteBending, Punching Shear design checks are performed at line support locations and the program automatically determines the punching shear perimeter (critical section). Additional information regarding punching shear design checks can be found on the [Punching Shear](#) page of the Help File.


## 2.4 Beams

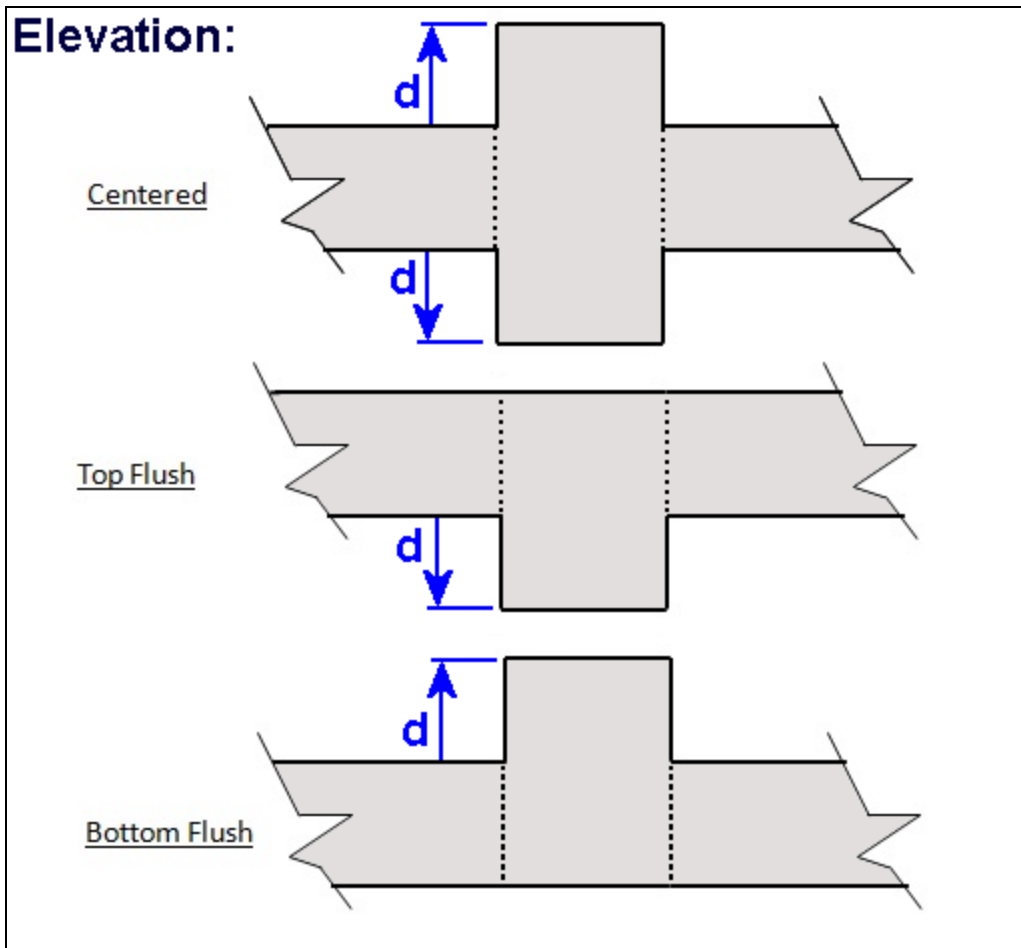
### Modeling

Beams are created in ConcreteBending by setting the drawing mode to "Draw Beams" from the **Ribbon | Boundary and Support** tab and sketching them on top of an existing slab in the model. Beams are used to distribute applied loads to the concrete slab and add stiffness to the model. Beams use the concrete compressive strength ( $f'c$ ) defined for the project, which can be found in the **Project Manager | Modify** tab when nothing is selected. Beams that adjoin one another, regardless of orientation, are modeled such that load transfer (shear and moment) occurs between the members (i.e. there are no end release options for beam elements).

### Properties

Beams are rectangular in shape and have a specified Beam Width and Projected Depth. All beams have a Vertical Offset Type and Length (defined by the start and end locations).

 The **Centered** vertical offset option uses **twice the depth** you specify, as shown below.



## Loading

Vertical forces, in addition moments in two directions can be applied to beams. The self weight of the beam can be included by checking the option for the selected beam in the **Project Manager | Modify** tab. Beam loads can be entered as distributed evenly along beam length or as the resultant total. Loads can be entered in the global coordinate system or in the beam's local coordinate system (defined as parallel or perpendicular to the beam).

## Analysis

In the finite element model, beams are modeled using member elements in the same plane as the plate elements used to model the slab. The stiffness of each beam is adjusted based on the Vertical Offset setting (i.e. the parallel axis theorem is used to modify the stiffness).

The results from the finite element analysis can be displayed graphically in the Analysis Results view. The **Project Manager | Results** tab displays the numerical results that correspond to the colored graphics. With nothing selected, the results displayed in the Project Manager are a summary for the selected result case, whereas if a single beam is selected, the Project Manager shows the result range for the selected beam. The various Result Cases that were included in the finite element analysis can be selected using the *Result Case* drop down from the **Ribbon | Home** tab. Furthermore, moment, shear, and displacement diagrams for each result case can be viewed using the [Beam Graph](#) tab. Analysis results can also be reported in tabular form using the [Text Reports](#).

## Design

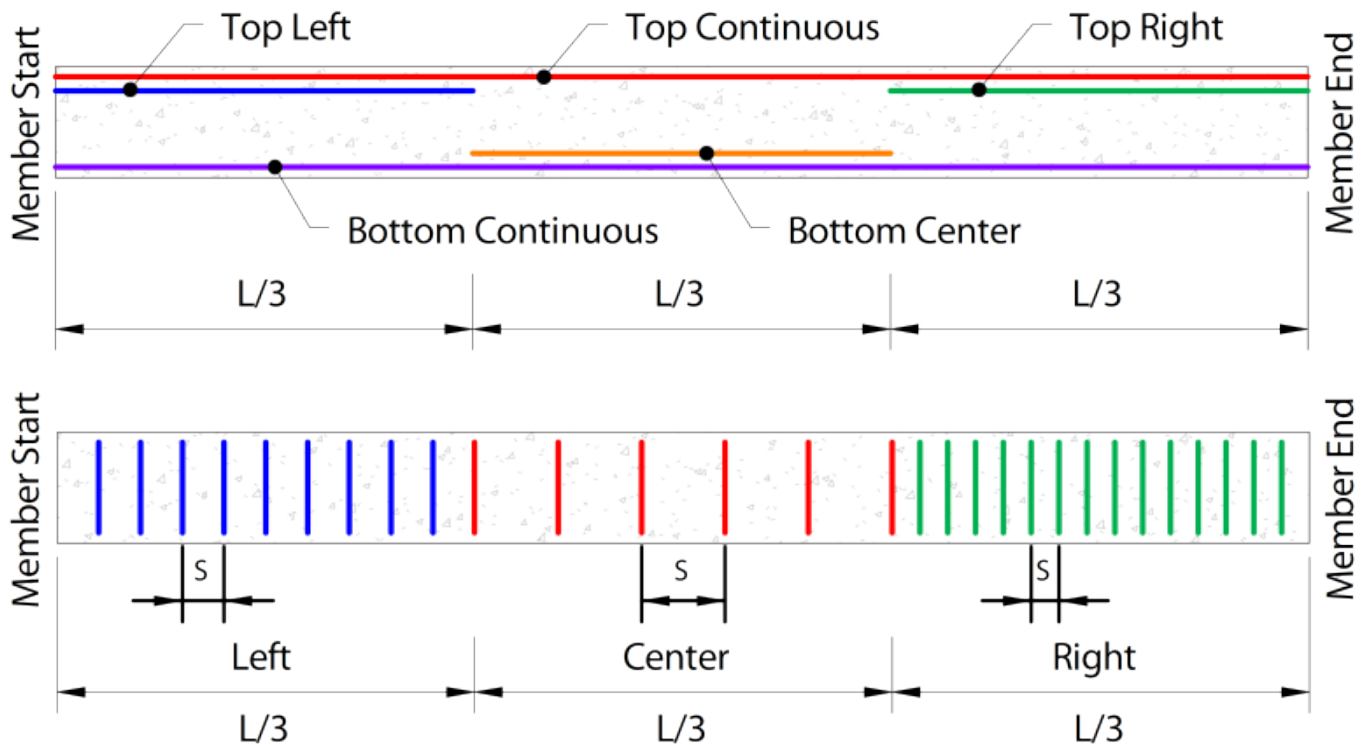
ConcreteBending checks and designs concrete beam members according to specifications listed below.



- ACI 318-19
- ACI 318-14
- CSA A23.3:19
- CSA A23.3-14

## Reinforcement Regions

Longitudinal and transverse reinforcing vary along the length of the member, as shown below (respectively).



## Beam Assumptions and Limitations

- Axial forces are not considered.
- Rebar is calculated for each  $1/3$  span. Rebar is assumed to be terminated at the  $1/3$ rd points of the beam, cutoff locations are not determined, and development length issues are not considered in the software.
- No "diagram" is produced to graphically display the bar locations or stirrup arrangements. The specified reinforcement for each beam can be displayed using Design Tables in the [Text Reports](#).
- Deep beams cannot be designed in ConcreteBending and a warning will be provided if a beam has a clear-span length less than  $4h$ .
- Beams are designed as rectangles, not T-Beams.

## Design Parameters

When designing beams, several parameters must first be defined. These parameters can be set by selecting a beam from the **Design Results** view and using the **Project Manager | Modify** tab.

**Beam Details Specification** - The Design Specification used to design the beam.

**Disable Checks?** - Causes selected design group to be omitted from design checks.

**High Seismic? - (ACI Only, Use Reduced  $\phi$  Factors for Members Resisting Earthquake Effects)**

Enabling this parameter lowers the  $\phi$  factors as indicated by ACI 318 Section 21.2.4 for members that are designed to resist earthquake effects and are part of a structure that relies on special moment resisting frames or special structural walls to resist earthquake effects. ConcreteBending relies solely on this parameter in determining whether or not to use reduced  $\phi$  factors (it does not attempt to calculate whether the shear capacity is greater than the shear corresponding to the development of the nominal flexural strength of the member). Only shear  $\phi$  factors are influenced by this parameter for design according to the ACI specification.

**Overstrength?** - Causes the member to be designed using overstrength load combinations.

**Check Level** - Determines the level of detail reported from design checks. Options are: *To Failure* (Fastest), *Each Limit State*, and *All* (Slowest, but provides the most information).

**Start/End Column Widths** - Widths of supporting columns at start and end of beam. These widths are used for determining where critical moment, shear, and torsion are at the ends of the beam. The critical demands are taken at the face of the column since member's effective depth significantly increases once the column is reached. These ends correspond to the member's local axes, where the local x-axis goes from the start-node to the end-node. Note that the critical section for shear can be taken "@ d" from the face of the support using the "Shear "@ d" from start/end" parameter (below).

**Shear "@ d" from start/end** - When enabled, the shear value calculated "@ d from the face" of the support is used for the shear and torsion checks when the check location is between the face of the support and d. Note that when using the CSA design specification, "dv" is used instead of "d".

**Reinforcement Details**

**Use Metric Bars** - Should metric reinforcement be used instead of imperial bar sizes?

**Longitudinal  $F_y$ :** Specified yield strength of the longitudinal reinforcement in the beam.

**Top, Bottom, Side (Torsional) Reinforcing**

**Top Reinforcement** - The size and quantity of Top reinforcement throughout the beam.

**Bottom Reinforcement** - The size and quantity of Bottom reinforcement throughout the beam.

**Side Reinforcement** - The size and quantity of Side reinforcement throughout the beam per each side. Note side bars are not used to resist flexure.

**Shear Reinforcement**

**$F_y$**  - Specified yield strength of the stirrups in the beam.

**Size** - The size of transverse reinforcement.

**Number of Legs** - The number of legs used for shear reinforcement.

**Are Stirrups Closed?** - Are closed stirrups used throughout the section?

**Spacing** - The center-to-center spacing of the stirrups. A different stirrup spacing can be specified for each 1/3 of the beam length. A spacing of 0 means no stirrups are provided.

**Beam Cover**

**Top Cover** - Concrete clear cover at the top of the section. Calculated as the distance from the top of the stirrup to the top of the section.

**Bottom Cover** - Concrete clear cover at the bottom of the section. Calculated as the distance from the bottom of the stirrup to the bottom of the section.

**Side Cover** - Concrete clear cover at the side of the section. Calculated as the distance from the side of the section to the stirrup.

**Displaying Design Results Graphically**

Beam results can be viewed graphically in the Design Results view. By default, the controlling unity value is displayed for beams. The unity results for a specific design check (such as the *Shear Check*, *Flexure Check*, *Torsion Check*, etc.) can be viewed by changing the *Design Information* parameter under the Beam Details category of the **Project Manager | Design Filter** tab.

### Reporting Beam Design Results

Design results can also be viewed in text form using the [Text Reports](#) view. Once on the Text Report tab, a list of available design tables is shown on the **Project Manager | Tables** tab and can be added to the text report by double-clicking an individual table or dragging and dropping a table into the report.

## 2.5 Load Combination Criteria

In ConcreteBending, strength level (LRFD) load combinations are required to perform concrete design checks. The analysis results from service level (ASD) load combinations can be viewed, but they are not used for design. Several sets of both ASD and LRFD building code load combinations are built into ConcreteBending. Custom load combination sets may also be created.

### Custom Load Combinations

Use the **Create Factored Combination** button located in the **Load Case Manager** to create any custom combination needed. Custom factored load combinations can be imported from the clipboard using the **Import From Clipboard** button in the Load Combinations tab in the **Load Case Manager**. Text must be tab delimited and copied to the clipboard in the following format:

```
{ComboName} {Factor} {ServiceCaseName} {Factor} {ServiceCaseName2} ...
```

For example:

ComboName	1.2	D	1.6	L	0.5	Lr
MyCombo	0.9	D	1.3	W		

### Environmental Design

Environmental design in accordance with ACI 350 requires analyzing all service cases with loads in them. Go to the Service Case tab of the Load Case Manager. Select the service cases with loads in them and check Include in Analysis.

### Seismic Criteria

In order to correctly generate load combinations that contain seismic loads, several additional parameters are required. Please refer to ASCE 7, Section 12.4, for how these parameters are used in generating load combinations. It is important to note that these parameters (such as SDS and SD1) are only used to generate load combinations. For example, ASCE 7 says that Seismic Category A does not require the combined orthogonal direction combinations (e.g. X+30%Y), but Category D does. ConcreteBending does not automatically generate any loads, just the combination cases.

## 2.6 Loads

In ConcreteBending, loads are applied to the model in a service load case. The appropriate service case can be selected in the Service Case drop-down list in the Home or Loads ribbon. Loads may be easier to see and select by rotating the view to get a 3D perspective of the model. Note: ConcreteBending only allows deflection in the Z-direction and rotation about the X-axis and Y-axis.

## Concentrated Loads

Concentrated loads are applied to [Load Points](#) by selecting the Load Point(s) and clicking the Apply Load button in the Loads ribbon. The concentrated load consists of a force in the global Z-direction and moments in the global X-direction and Y-direction.

## Line Loads

Line loads are applied to [Beams](#) by selecting the Beam(s) and clicking the Apply Load button in the Loads ribbon. The line load consists of a force in the global Z-direction and/or moments which can be defined in the global X-direction and Y-direction or defined parallel and perpendicular to the beam (i.e. in the beam's local direction). Line loads can be entered as resultants or as distributed loads in the model.

## Area Loads

### Partial Slab Boundary Loads

Rectangular, circular, tubular, and ring loads are used to apply pressure loads and/or overturning moments to some portion of the [Slab](#) in the model. Pressures may be uniform or linearly varying in the global X-direction or global Y-direction. First set the Area Load type in the Loads ribbon and then sketch the load on the drawing grid in the model. The size and location of the load can be modified if needed after the load has been generated. Note: Loads will not be applied to holes in the slab's boundary.

### Complete Slab Boundary Loads

Select one or more concrete [Slab\(s\)](#) and use the Apply Load command in the Loads ribbon to load the selected slab(s). Alternatively, the Apply Multiple Boundary Load command in the Loads ribbon can be used to create a load that is applied to all boundaries in the model. Note: Slab boundary loads will automatically adjust to load the entire boundary if the boundaries's size or location is modified.

## 2.7 Self Weight

### Project Type

Two basic types of models can be created in ConcreteBending, one where gravity acts perpendicular to the concrete boundaries (e.g. elevated slabs) or one where gravity is in the plane of the concrete boundaries (e.g. vertical walls). This setting can be changed in the **Project Manager | Modify** tab from the **Model and Load** view with nothing selected using the *Self Weight* drop-down menu. In a "Slab" model, the self weight of the elements can be set on an individual basis, whereas in a "Wall" model, the self weight of the elements is ignored. The project setting takes precedent over the self weight setting on any individual model object.

### Slabs and Beams

To include a slab or beam's self weight in the Dead-Load load case, select the object, switch to the Model and Load view and check the Add Self Weight box in the **Project Manager | Modify** tab. The item's density is determined by the selected material.

### Point Supports, Line Supports, and Load Points

These items do not have self weight

## 2.8 Load Points

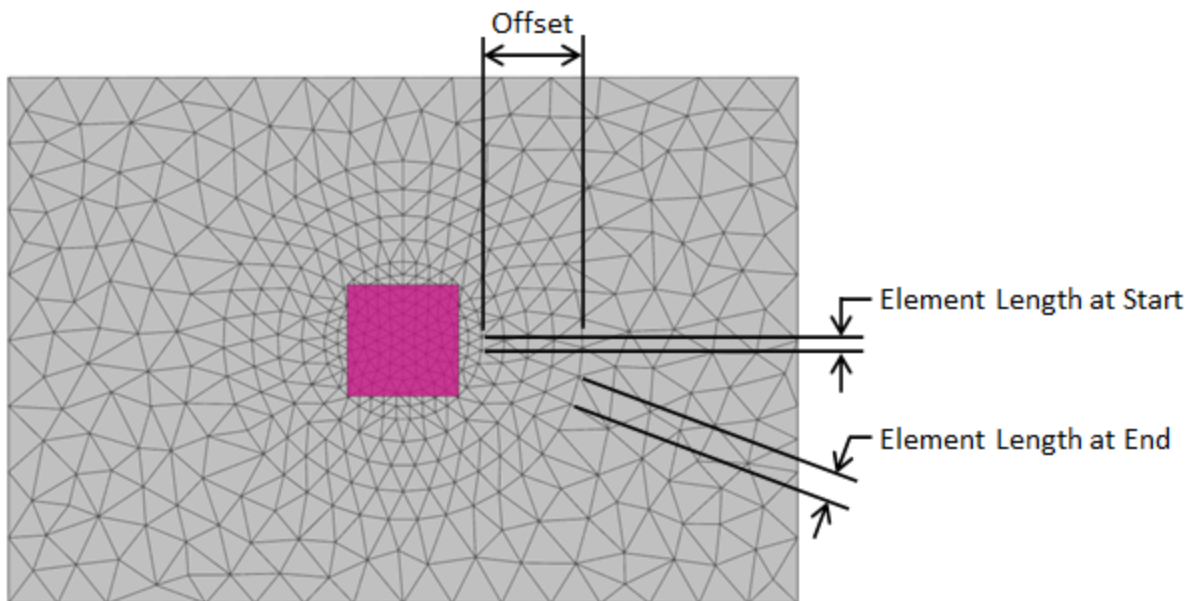
Load Points in ConcreteBending are used to apply concentrated loads and moments to slabs at a specified location.

### Modeling

Load points are defined by their location, shape, and size in the model. Four shapes are available: Square, Rectangular, Circular and Octagon. The radius of an octagon pier is based on the octagon being inscribed inside a circle with the specified radius. Note: In the Finite Element Analysis (FEA) model, a single node in the slab's mesh is loaded; therefore, care should be taken when specifying load points with large cross-sections.

### Mesh Refinement

The FEA method is approximate and the accuracy of the FEA solution typically increases as the mesh becomes finer. Generally, a finer mesh should be used in regions where there are large changes in stresses in the plates (such as at load points where concentrated loads can create stress concentrations). Select a load point(s) and set Refine? = True in the **Project Manager | Modify** tab to refine the mesh in the regions of the chosen load point(s). The mesh refinement parameters for load points are defined in the image below.



**Load Point Mesh Refinement**

### Loading

Concentrated forces (compression and tension), in addition to moments in two directions can be applied to load points.

### Analysis

During the finite element analysis each load point is modeled as a single node connected to the slab's finite element mesh. Therefore, care should be taken when specifying load points with large cross-sections.

### Design

While load points appear to be columns when the Picture View is enabled in ConcreteBending, they do not have any

# ConcreteBending 8.0 User's Guide

length in the finite element model (i.e. they are not modeled as a member element). Therefore, load points are not designed in the program. Punching shear design checks, however, are checked for the slab at the location of the Load Points.

## Punching Shear

In ConcreteBending, Punching Shear design checks are performed at load point locations and the program automatically determines the punching shear perimeter (critical section). An individual load point can also define punching shear reinforcement in the form of headed shear studs or shear stirrups located in the slab. Additional information regarding punching shear design checks and punching shear reinforcement can be found in the [Punching Shear](#) section of the Help File.

## Report

Multiple report tables, which include modeling, loading, and result information for the Load Points, are available in the Text Report view.

## 2.9 Analysis

ConcreteBending uses finite element analysis (FEA) to determine the displacements, shears, and moments in the slab system. Exporting the model as a VisualAnalysis project (**File | Export a VA Project**) will show what is happening behind the scenes in ConcreteBending. While ConcreteBending attempts to handle many details of the FEA method, some knowledge of FEA is still required to use the program successfully.

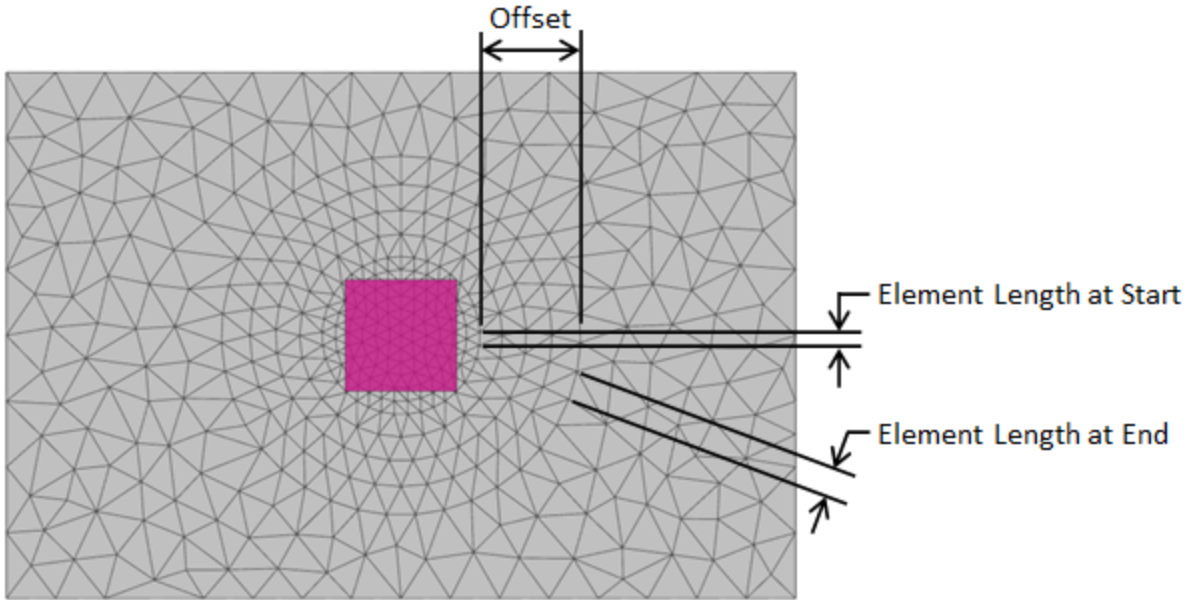
## Mesh Settings

To control the number of plate elements, pick between course, medium, fine, or specify a desired number directly in the **Project Settings**. The actual number of meshed plates generated in the model is shown under Meshed Plates. Turn on Meshed Plates in the **Project Manager | Filter** tab to view the meshed plates in the model. In ConcreteBending, the advanced finite element meshing options can be modified if needed. Note: The mesh settings at [Load Points](#) and [Point Supports](#) can be set by selecting the model object(s) and adjusting Mesh Refinement settings in the **Project Manager | Modify** tab.

## Mesh Refinement

It is the user's responsibility to verify and validate the results obtained from ConcreteBending. The finite element method is approximate, and the accuracy of the solution depends on how fine the mesh is in the model (generally, a finer mesh produces more accurate results). A finite element mesh refers to the multiple plates that are used to model a single component (such as a slab). Mesh Refinement is the process of reanalyzing the model with successively finer and finer meshes and comparing the results between these different meshes. As the mesh is refined, the change in the solution becomes smaller and an asymptotic behavior of the solution starts to emerge as shown in the figure under Mesh Refinement Procedure below. Eventually, the changes to the solution will be small enough that engineering judgment can be used to determine that the model has converged.

In ConcreteBending, mesh refinement is accomplished by reducing the Element Count in the **Project Manager | Modify | Project Settings**. Furthermore, as shown in the image below, the mesh can be refined at [Load Points](#) and [Point Supports](#) by selecting the model object(s) and adjusting Mesh Refinement settings in the **Project Manager | Modify** tab. In general, a finer mesh should be used in areas where there are large changes in stresses in the plates (e.g. load point locations that receive concentrated loads and point support locations where the support at a single node can create stress concentrations).

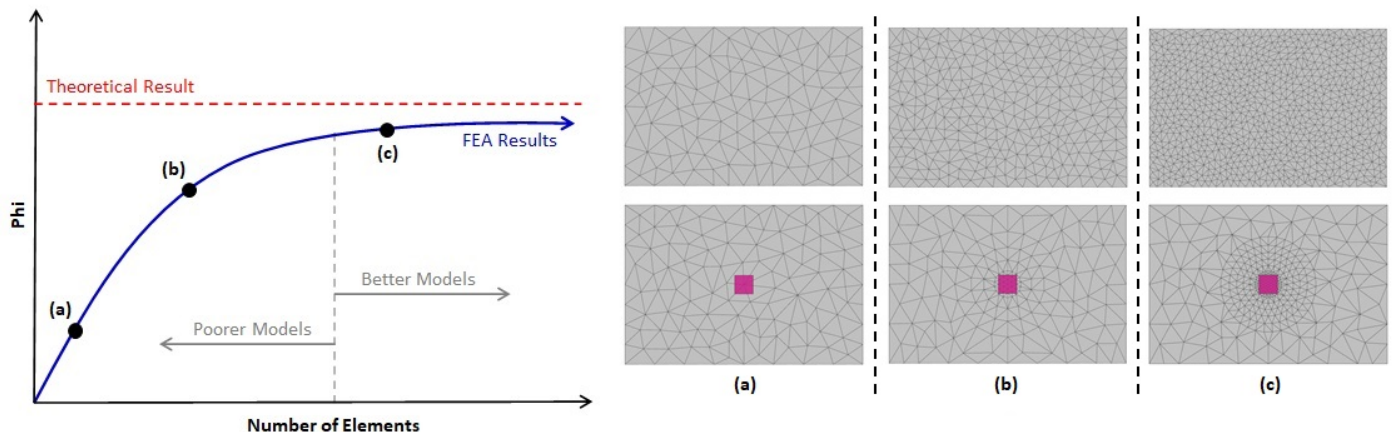


**Load Point & Point Support Mesh Refinement**

### Mesh Refinement Procedure

1. Model the slab and ConcreteBending will automatically generate the plate elements based on the Mesh Refinement settings.
2. Let the analysis run and record the results.
3. Increase the number of Meshed Plates for the slab and/or refine the mesh at Load Points and Point Supports as discussed above.
4. Let the analysis run and record the results.
5. Compare the results from Step 4 with the results from Step 2. If the difference in the analysis results is small and acceptable (using engineering judgment), the mesh refinement process is complete. If the difference in the analysis results is large and unacceptable (using engineering judgment), start back at Step 3.

Note: Model results such as displacement, moment, shear, etc., which are represented as Phi in the graph below, will coverage at different rates. Therefore, it is important to ensure that the model has converged for the result of interests.



### Plate Bending



# ConcreteBending 8.0 User's Guide

The bending part of the FEA plate element is based on the triangle formulation originally presented by Xu et. al.<sup>1</sup> in 1992. This element accounts for transverse shear effects present in structures that might contain areas with thick plates, such as footings or thick floor slabs.

## Statics Checks

A Statics Check is performed for each load case analyzed in ConcreteBending. The total applied loads in each global direction is calculated and compared to the sum of all support reactions in the corresponding global directions. The applied loads are based on the deformed shape of the structure while the reactions are based on the structure's undeformed shape. If the loads and reaction are equal and opposite in magnitude, then the structure is in equilibrium. An imbalance indicates that the deflections are large enough to generate inaccurate results which might indicate that there is a modeling problem. ConcreteBending provides a warning if a significant imbalance is detected. If a warning is received, carefully review the model to ensure it is set up correctly and verify the results. The Statics Check is displayed on the **Project Manager | Results** tab or the Statics Check Information table can be added to the report.

## References

1. Xu, Zhongnian, "A thick-thin triangular plate element" *International Journal for Numerical Methods in Engineering*, Vol. 33, 1992, pp. 963-973.

## 2.10 Punching Shear

Punching (two-way) shear checks are performed at locations where concentrated loads are introduced into the slab, such as at load points, point supports, and line supports. ConcreteBending is capable of checking unreinforced punching shear for individual objects (Figure 1), reinforced punching shear for individual objects (Figure 2), and unreinforced group punching shear (Figure 3). For the design to pass, the strength of the critical section must exceed the net load acting inside the of the section.

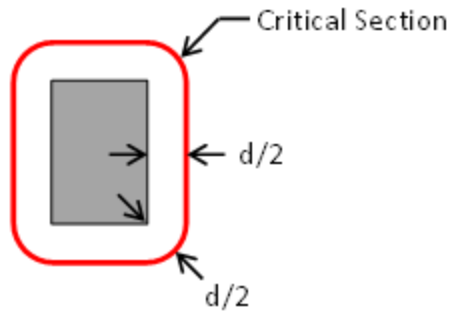
### Critical Sections

The design check for punching shear involves calculating the Critical Section around the object(s), which may be truncated by the slab boundary. The Critical Section is positioned so that the perimeter ( $b_o$ ) is minimized but is not located closer than  $d/2$  to the edges or corners of the object(s), as illustrated in Figures 1, 2, and 3. When punching reinforcement is provided, the design check is performed on both the Inner and Outer Critical Sections (Figure 2). Note: When the CSA design specification is selected, the Outer Critical Section does not extend  $d/2$  beyond the punching reinforcement, as limited by Clause 13.3.7.4.

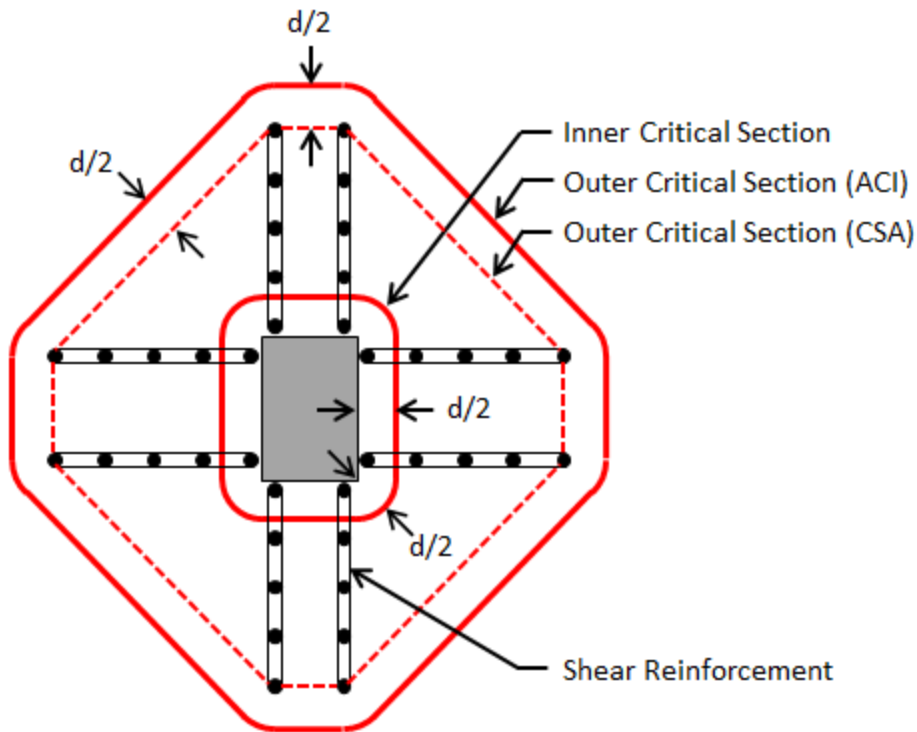
If a Beam crosses an object's Critical Section, punching shear checks are not performed for that object, and it is excluded from group punching shear. In this case, it is assumed that the Beam will be designed to handle the required shear demands. Note: "-N.A.-" will appear in the punching shear report for any objects excluded from the punching shear check. Furthermore, if a Beam crosses the outer perimeter of an object with punching reinforcement, only punching at the Inner Critical Section is checked. Note: Punching shear checks are not performed for loads applied directly onto the slab (e.g., rectangular, circular, tubular, or ring loads).

The Critical Section can be shown graphically in the Design Results view. When punching reinforcement is present, the Outer Critical Section is displayed graphically for the object with reinforcement only when punching at the Outer Critical Section controls over punching at the Inner Critical Section.





**Figure 1: Individual Punching Shear Critical Section - Unreinforced**



**Figure 2: Individual Punching Shear Critical Sections – Reinforced**

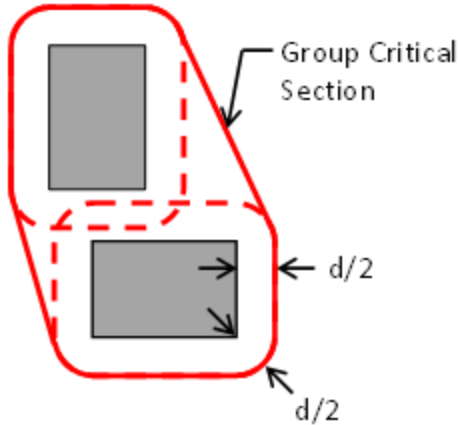


Figure 3: Group Punching Shear Critical Section –Unreinforced

## Reinforcement Depth

The Reinforcement Depth ( $d$ ) used in the punching shear calculation is dependent on the load direction. For loads on the slab in the positive Z-direction (e.g., point supports in compression, line supports in compression, and load points in tension),  $d$  is measured from the bottom of the slab. For loads on the slab in the negative Z-direction (e.g., point supports in tension, line supports in tension, and load points in compression),  $d$  is measured from the top of the slab. In ConcreteBending,  $d$  is measured to the center of the bars in the reinforcement mat that minimize its value (i.e., the size and orientation of the bars in the mat affect the value of  $d$ ). Figures 4 and 5 provide examples of how  $d$  is calculated for point supports and load points in both tension and compression, for single and double mat bar configurations, respectively. Note: Since ConcreteBending does not design the reinforcement between the various objects (point supports, line supports, load points, etc.) and the slab, the user must determine if the critical sections for punching of objects in tension are valid.

When the **Specify** design approach is used for the slab's reinforcement,  $d$  is calculated based on the specified reinforcement bar diameters. When the **Optimize** design approach is used,  $d$  is calculated for each plate element within the critical section using the suggested reinforcement for the design, and the minimum value is used for the punching shear calculation.

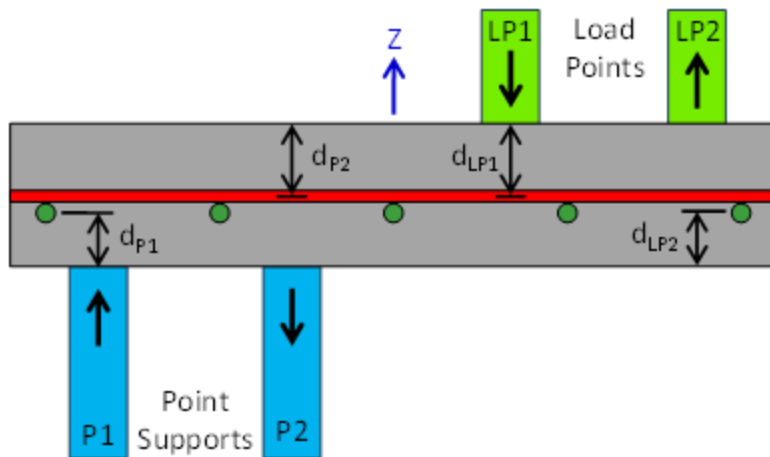
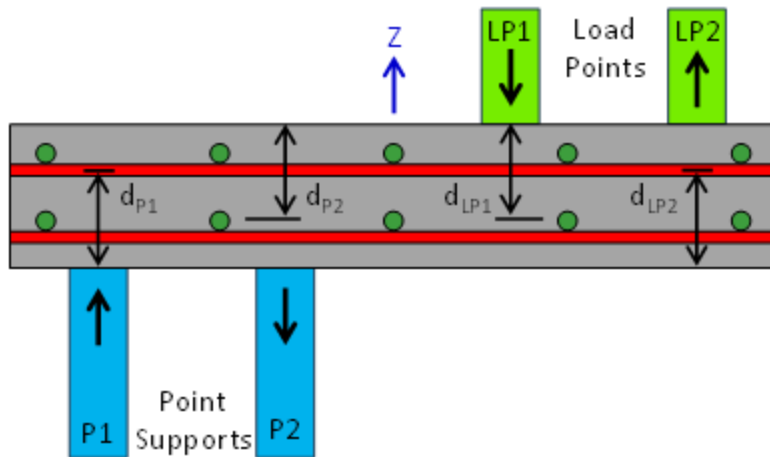


Figure 4: Reinforcement Depth ( $d$ ) for Punching Shear - Single Mat



**Figure 5: Reinforcement Depth ( $d$ ) for Punching Shear - Double Mat**

#### **Punching Shear Groups**

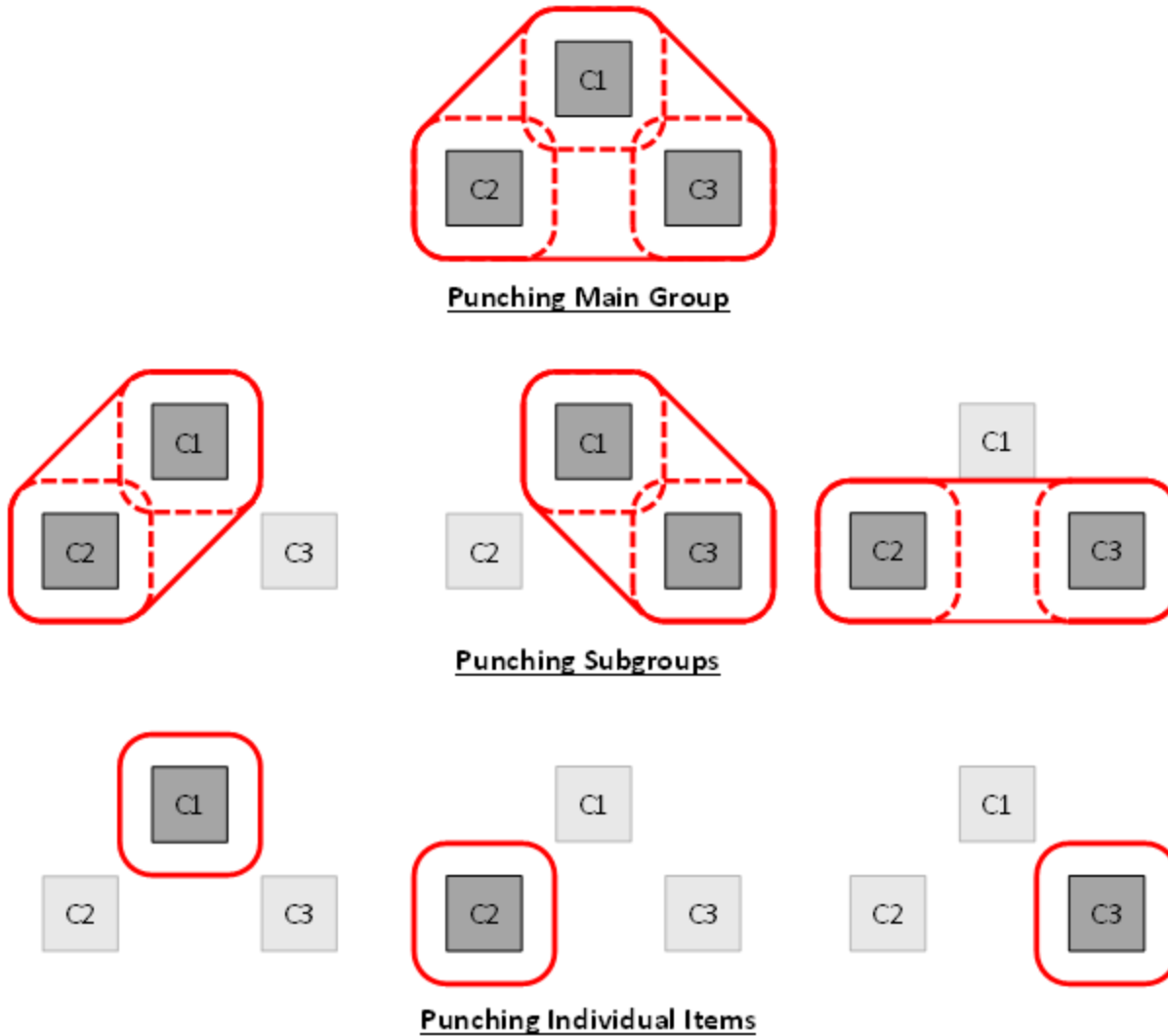
When Load Points, Point Supports, and Line Supports are closer to each other than the thickness of the slab, ConcreteBending combines them to form a punching shear group. These groups expand to include any additional elements within their boundary (e.g., if an item falls within the reentrant corner of a punching shear group, it will be incorporated into the group). For the group punching design check to pass, the strength of the group's critical section must exceed the net load acting inside of the group's critical section. To include Line Supports in the automatically generated punching design groups, the **Include Line Supports in Groups** parameter must be enabled and the **Line Aspect Ratio** must be set. Line Supports with a length/thickness ratio greater than the value specified will not be included in punching shear groups. Line Supports with large aspect ratios are unlikely to punch and may create undesirable groups.

Group critical sections can be shown graphically in the Design Results view. Although the group critical section cannot be directly overridden or modified, adjusting the **Override Perimeter Offset** parameter (see below) for an individual object within the punching shear group will affect the overall critical section for the group.

#### **Punching Shear Subgroups**

When a group contains more than two objects, the punching behavior of a sub-group may govern the design, rather than the behavior of the entire group or the individual elements within it (see Figure 6). For the subgroup punching design check to pass, the strength of the subgroup punching perimeter must exceed the net load acting within the subgroup perimeter. To account for this, enable the **Check Subgroups** parameter and set the **Max Group Size** parameter. Note: Groups with more items than the specified number will not be checked for subgroup punching.

The number of subgroups checked grows exponentially, following the equation  $2^n - n - 2$ . For example, a group of 3 objects will have  $2^3 - 3 - 2 = 3$  subgroups (in addition to 1 main group and 3 individual objects), and a group of 4 objects will have  $2^4 - 4 - 2 = 10$  subgroups (plus 1 main group and 4 individual objects), etc. Due to the exponential increase in computational cost as the group size grows, the **Max Group Size** parameter is capped at **10**. This ensures that the user does not request a problem that exceeds realistic computational time limits.



**Figure 6: Example of All Possible Critical Sections for a Group of 3 Items**

**Punching Reinforcement**

Two-way shear reinforcement can be defined for individual Load Points or Point Supports in the form of headed shear studs or shear stirrups. When punching reinforcement is provided, a design check is performed on both the Inner and Outer Critical Sections. The slab's capacity at the Inner Critical Section is calculated as the sum of the concrete and steel reinforcement capacities, while the slab's capacity at the Outer Critical Section is determined based solely on the concrete capacity. Note: The controlling critical section is displayed graphically in the Design Results view.

Note: The punching shear reinforcement is assumed to extend away from the object in the Global +X, +Y, -X, and -Y directions, regardless of the object's shape or orientation. If a slab edge (either an outside edge or hole edge) is located within the Reinforcement Offset distance (see below), the reinforcement is assumed to extend only to the edge of the slab in that direction.

**Punching Design Parameters**

Below are the group punching shear design parameters. These parameters can be found from the **Design Results | Modify** tab.

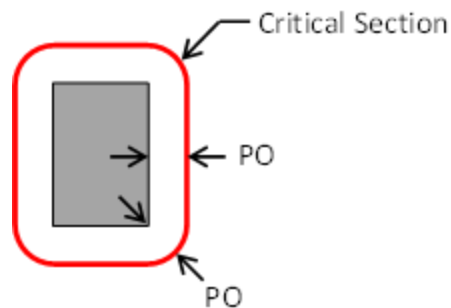
**Group Punching Shear**

- Include Line Supports in Groups?** Do you want to include Line Supports in group punching shear? - *Choosing 'Yes' allows you to set the Line Aspect Ratio parameter. Line Supports with a length/thickness ratio greater than specified will not be included in punching shear groups. Line Supports with large aspect ratios are unlikely to punch and may create undesirable groups.*
- Check Subgroup Punching?** Do you want to check subgroup punching? - *Choosing 'Yes' allows you to set the Max Group Size parameter. Punching groups containing more items than specified will not be checked for subgroup punching. Increasing this number incurs an exponential computational cost.*

Below are the punching shear design parameters available for an individual Load Point or Point Support object. These parameters can be found from the **Design Results | Modify** tab when a Load Point or Point Support is selected.

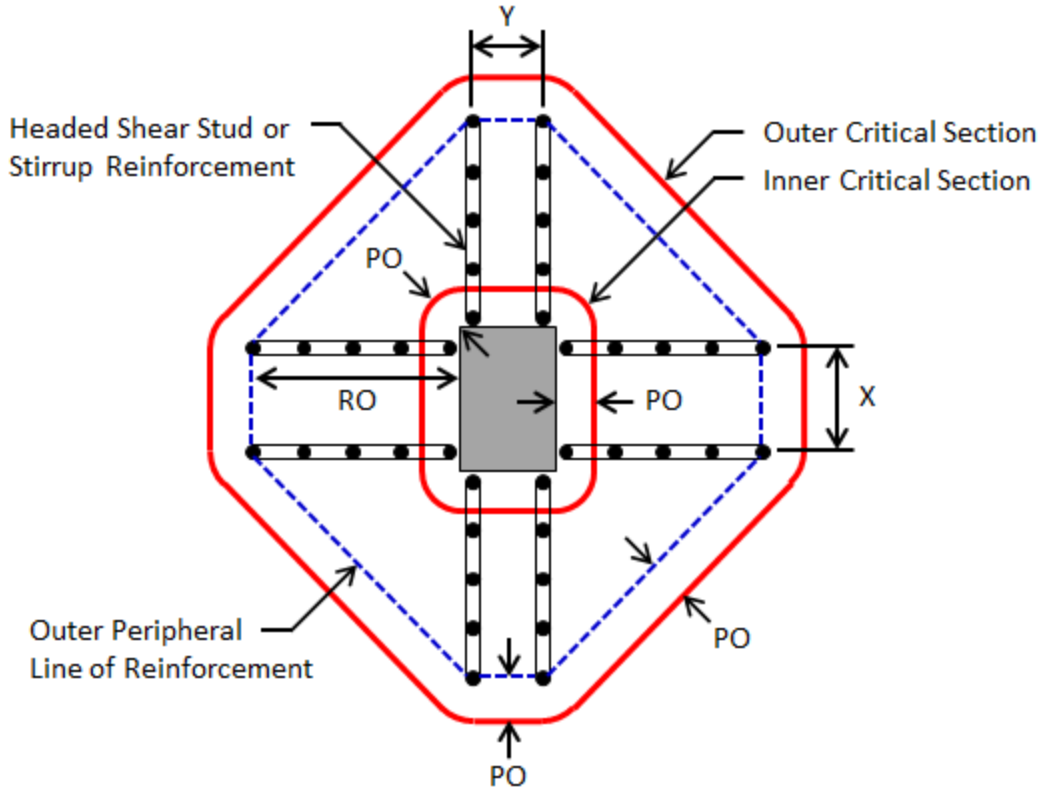
## Load Points & Point Supports Without Reinforcement

- Override Punching Offset (PO)?** Do you want to override the punching perimeter offset? - *Choosing 'No' results in using the slab reinforcement depth ( $d/2$ ) for the offset. Choosing 'Yes' allows you to enter your own value to be used as the punching perimeter offset.*



## Load Points & Point Supports With Reinforcement

- Override Punching Offset (PO)?** Do you want to override the punching perimeter offset? - *Choosing 'No' results in using the slab reinforcement depth ( $d/2$ ) for the offset. Choosing 'Yes' allows you to enter your own value to be used as the punching perimeter offset.*
- Are Headed Studs?** Is the reinforcement headed shear studs?
- Fyt** The yield stress of the reinforcement.
- Av total** The total area of shear reinforcement around a peripheral line of reinforcement.
- Spacing** The spacing of peripheral lines of shear reinforcement.
- X Reinf. Group Width (X)** The width of the stud/stirrup reinforcement group oriented in the Global X-Direction. Enter 0.0 for a single stud rail or stirrup leg line.
- Y Reinf. Group Width (Y)** The width of the stud/stirrup reinforcement group oriented in the Global Y-Direction. Enter 0.0 for a single stud rail or stirrup leg line.
- Reinforcement Offset (RO)** The distance from the object face that the shear reinforcement extends outward.



## 3 Report

### 3.1 Reports

Reports in ConcreteBending are designed to present information in a clear, concise, and organized fashion. Reports can include both text-based and graphical information that can be printed to paper, to .pdf, or saved in a number of different file formats. Graphical information can be inserted into a report using the **Copy** and **Paste** commands or printed directly using the **File | Print** command.

#### Report Essentials

- [Tables](#)
- [Saved Styles](#)
- [Beam Graphs](#)

#### Custom Report Logo

The report may be customized to include your own (company) logo in the header. All you need to do is create a logo image: ReportLogo.png or ReportLogo.jpg, and place it in the IES\Customer folder, which you can access via the **Tools | Custom Data** toolbar command. The image should be kept to less than 5 times wider than it is tall. It will be scaled to fit in the header area, but wide images may cause other text to start wrapping or get truncated. If the image works you'll see it in the report/preview immediately after restarting ConcreteBending. This feature is also available in other IES tools.

### 3.2 Tables

In ConcreteBending, tables are used to report information in a clear and concise manner. The tables available for the report are listed in the **Project Manager | Tables** tab when the Report View is active. Tables fall into one of five categories (Project, Structure, Load, Result, and Design) and will automatically appear or disappear depending on the items in the model (elements, loads, etc.) and the available analysis and design results. Hover the mouse over a table in the list to view its description.

#### Table Types

- **Project Tables** are used to document the project wide information for the model including the Project Settings, Service Load Cases, and Factored Load Combinations.
- **Structure Tables** are used to document the input data for various model objects including Slabs, Load Points, Point Supports, Line Supports, Beams, etc. Also, a Model Summary can be reported.
- **Load Tables** are used to document every load applied to the model in each service load cases including Area Loads, Point Loads, Beam Loads, etc.
- **Result Tables** are used to document the analysis results for the elements in the model including Plate Forces and Displacements, Beam Forces and Displacements, Point and Line Support Results, etc.
- **Design Tables** are used to document the Design Settings, Design Summary, Shear Design, Punching Shear Design, Slab Flexure Design, Beam Design, etc. These tables are used to document the demand, capacity, unity checks, show intermediate calculation values, controlling code references, etc.

#### Adding & Removing Tables

To add a table to the report, simply **drag** the table from the **Project Manager | Tables** tab to the desired location in the report or **double-click** on the table to insert it at the end of the report. A list of the report's Included Tables is shown in

# ConcreteBending 8.0 User's Guide

the **Project Manager | Tables** tab which can be rearranged by *dragging* them with the mouse. To remove a table from the report, click the X next to the table in the Included Tables list or *right-click* on the table in the report and select Remove.

## Modifying Tables

Tables can be modified using the Report Settings or Model Filters in the **Project Manager | Report Filter** tab. The Report Settings are used to specify which Service Cases and Result Cases to include in the report while the Model Filters are used to filter the items that are included in the report (such as Slabs, Point Supports, Load Points, Beams, etc.). Tables can also be modified by clicking on the tables in the report. *Click* the column header to sort the column, *drag* the column header to rearrange the columns in the tables, or *drag* the column borders to adjust the column widths.

### Selected Table

*Click* within a table to select the table and activate the Selected Table section in the **Project Manager | Report Filter** tab. In this tab, the Title can be modified, the columns can be sorted, and the page width can be defined. Choose which columns are included in the table under the Columns section and drag the columns in this section to rearrange them in the report.

### Selected Table Extremes

Certain tables have the Selected Table Extremes option available in the **Project Manager | Report Filter** tab. The following parameters are used to set how the information is filtered in the selected table.

- **Extreme Rows** - Set to show the Extreme Rows Only for the table or to Show All (which can lead to lengthy reports that may need to be filtered by result cases or reported items to be manageable).
- **Included Rows** - Specify how the extreme rows are considered.
  - **Max and Min** - Keep only the max and min values.
  - **Max** - Keep only the max value.
  - **Min** - Keep only the min value.
  - **Max/Min (when opposite sign)** - Keep the max and min values, if different signs, else keep the most extreme.
  - **Extreme** - Keep only the most extreme value, positive or negative.
- **Applies To** - Specify if the extreme rows be kept on a table wide basis or by each item in the table.
- **Consider Zero as Extreme** - Specify if zero should be considered an extreme value.
- **Show All Extreme Rows** - Choose to show all rows with the extreme value or only show the first occurrence of the extreme value.

### Column Justification

Right click on a column in the report to set the Column Justification to Left, Center, or Right for an individual column in a table. In the Reports category of the Preferences, the default Justification for the Text data and Physical data can be specified.

## 3.3 Saved Reports

Both Project Reports and Report Styles can be created and saved in ConcreteBending. While Project Reports are saved in the .cbp project file, Report Styles are saved in the Custom Data Folder and can be used for multiple Projects. ConcreteBending also includes a few default IES Report Styles.

## Project Reports



## Save a Project Report

After creating a report, go to the **Project Manager | Reports** tab, name the report, and click the Save in Project button. This saves the Project Report in .cbp project file for easy access.

## Delete a Project Report

To delete a Project Report, click the X next to the report in the **Project Manager | Reports** tab. The **Home | Undo** command can be used to restore a deleted report.

## Report Styles

### Save a Report Style

After creating a report, go to the **Project Manager | Reports** tab, name the report, and click the Save as Style button. This saves the Report Style in the Custom Data Folder in an XML file and makes the style available for use in other ConcreteBending projects.

### Create a Report from a Style

To create a Report from Report Style, Simple *double-click* on the saved report in the **Project Manager | Reports** tab or *drag* the saved report onto the Report View.

### Update a Report Style

To update a Report Style, create a report based on the style, modify the report as need, and save the style using the original name.

### Delete a Report Style

To delete a style, click the X next to the style in the **Project Manager | Reports** tab or manually delete the style in the Custom Data Folder in an XML file. The only way to restore a style once it is deleted is to import a backup copy of the style file.

### Share Report Styles

Report Styles can be shared by copying the XML style file (found in **Tools | Custom Data**) to the same location on another computer. The XML file can be manually edited to merge styles from other users. Always save a back up copy of the XML file before doing any manual customization so the file can be restored if it gets corrupted.

## 3.4 Beam Graphs

Beam Graphs display detailed diagrams (displacement, moment, shear, and torsion) for beams along their length. The results can be displayed for one single beam or for a chain of beams for a single Result Case.

### Create a Single or Multi Beam Graph

To create a single beam graph, simply select a beam in the Model and Load, Analysis Results, or Design Results view and switch to the Beam Graph view. A different beam can be chosen from the dropdown in the **Project Manager | Graph Filter** tab. To create a multi beam graph, simply select two or more connected beams that form a line and switch to the Beam Graph view. If the selected beams do not have local axes in the same direction, a note will appear on the graph indicating that the member chain has inconsistent local axes.

## Customize a Beam Graph

Beam graphs can be customized in the **Project Manager | Graph Filter** tab. Annotations, Data Points, Grid Lines, and Shadows can be adjusted for the plots. Use the Details Dialog to change specific graphic format information like colors and fonts as well as the basic type of plot used.

## Print a Beam Graph

Beam graphs can be printed directly using the **File | Print** command. The orientation (portrait or landscape) of the graphs can be set using the **File | Page Setup** dialog. Use **File | Print Preview** to view the page before printing.

## Export a Beam Graph

To export a Beam Graph to another program, use the **Home | Copy** command to copy the Beam Graph to the clipboard and then use **Paste** to insert the picture into the other application.

## 4 Integration

### 4.1 IES VisualAnalysis

#### VisualAnalysis Files (.vap) Created by ConcreteBending

ConcreteBending can be used to create a VisualAnalysis project file (.vap) using the **File | Export a VA Project** feature. The .vap file can be used to look at the finite element model that was created behind the scenes in ConcreteBending or to perform more advanced analyses (such as a dynamic analysis) which are not supported in ConcreteBending. Note: [VisualAnalysis](#) is licensed separately by IES. [Contact Sales](#) for additional information regarding this product.

### 4.2 IES VAConnect

#### Connection Design (Export from ConcreteBending to VAConnect)

ConcreteBending can export point loads to VAConnect for a steel-column base plate design. After defining all the loads, select the load point(s) in the Model & Loads view, and use the **Boundary and Support | Export to VAConnect** feature. Note: [VAConnect](#) is licensed separately. [Contact Sales](#) for additional information regarding this product.

## 5 Script

### 5.1 Script Overview

#### Introduction

The script feature in ConcreteBending is a powerful tool used to create models objects, generate service cases, apply loads, extract results, etc. using a command line interface. In addition to supporting the various [Commands](#) outlined in this Help File, the command line will accept any valid command in the [C# Programming](#) language (allowing the use of if statements, for loops, etc.). In addition to using the command line directly, more complex [Scripts](#) can be generated in any text file and read into the program. This allows models with complex geometry to be created, parametric studies to be performed, finite element meshes to be automatically refined until convergence, etc.

#### Command Line Basics

##### 1. Location

- a. The Script tool is accessed by selecting the Script tab at the bottom of the Find Tool. By default, the Script will replace the Find Tool when selected, but the two can be floated and docked independently if needed.



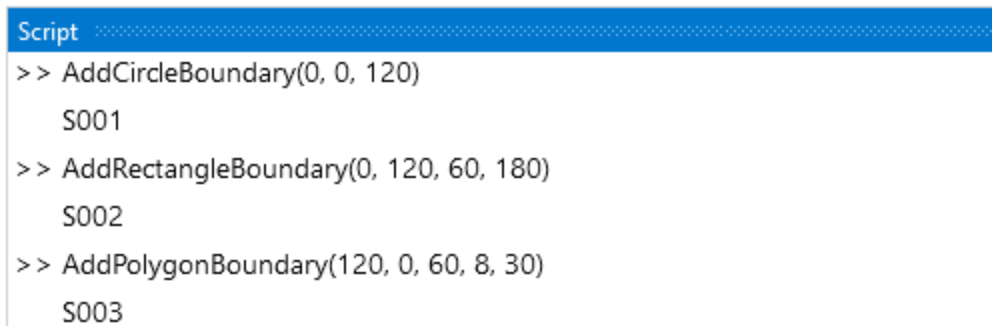
##### 2. Units

- a. Input and output are always in the current unit style. The style can be changed from the command line.



##### 3. Add Boundaries

- a. Add circular, rectangular, or polygon Boundaries of specified sizes and location using the appropriate commands.



- b. The return value can be suppressed by using a semi-colon on at the end of the command.

## Script

```
>> AddCircleBoundary(0, 0, 180);
>>
```

- c. Information can be stored in variables and math can be performed in the command line. Variables circumvent the need to input the same number repeatedly. The example below uses variables to defined the center coordinates and the width of a rectangular boundary (assuming the current unity style is Kips & Inches). Math is then used in the in the command line to define height of the boundary as two times the width. Note: When storing a value in a variable, the line must end with a semicolon.

## Script

```
>> var x = 72;
>> var y = 144;
>> var w = 24*12;
>> var h = 2*w;
>> AddRectangleBoundary(x, y, w, h);
```

- d. Boundaries of non-standard shape can be defined based on lists of the X and Y coordinates or based on an array of Locations. Note: Locations can be stored as variables and used to define various items in the model, allowing the items to perfectly coincide.

## Script

```
>> var xCoordinates = new List<double>(){0, 120, 0};
>> var yCoordinates = new List<double>(){0, 0, 120};
>> AddBoundary(xCoordinates, yCoordinates);
>> var location1 = new Location(0, 0);
>> var location2 = new Location(-120, 0);
>> var location3 = new Location(0, 120);
>> AddBoundary(location1, location2, location3);
```

- e. The slab's reinforcement details (e.g. bar configuration, bar cover, design approach, etc.) can be defined using the various commands.

## Script

```
>> SingleMat_Specify("S1", true, 2, "#6", 12, "#8", 16)
    Modified
>> DoubleMat_Optimize("S2", true, 2, true, 3, ("#4", 12), ("#5", 12))
    Modified
```

#### 4. Add Point Supports

- a. Point Supports can be added to the project at specific coordinates or Locations.

## Script

```
>> AddPointSupport(0,0)
    P001
>> var location = new Location(12, 24);
>> AddPointSupport(location)
    P002
```

- b. Point Supports can be renamed if the name that is automatically generated is not satisfactory.

```
Script
>> NamePointSupport("P001", "P1")
Modified
```

- c. Once the Point Support is added, it will automatically be selected, and its properties can be modified in the Project Manager as usual. Alternatively, there are various commands to modify the Point Support parameters, see the [Commands](#) page for a complete list. Note: Item names must be enclosed in quotes.

```
Script
>> MovePointSupportTo("P1", 12, 24)
Modified
>> RectanglePointSupportSize("P1", 6, 12)
Modified
>> FixPointSupport("P1", "DZ")
Modified
>> RefinePointSupport("P1", 6, 0.5, 4)
Modified
```

## 5. Add Beams/Line Supports

- a. Beams and Line Supports can be added to the project based on start and end coordinates or locations.

```
Script
>> AddBeam(0, 0, 120, 240);
>> var startLocation = new Location(0, 0);
>> var endLocation = new Location(-120, -240);
>> AddLineSupport(startLocation, endLocation);
```

- b. Various properties can be set when creating Beams or Line Supports.

```
Script
>> AddBeam("BM1", 0, 0, 120, 240, 24, 12, true);
>> AddLineSupport("LS1", 0, 0, -120, -240, 12);
```

- c. A variety of parameters can be modified for existing Beams or Line Supports.

```
Script
>> SetBeamTopFlush("BM1");
>> FixLineSupport("LS1");
```

- d. Beams and Line Supports can be moved using the command line.

```
Script
>> MoveLineTo("BM1", 12, 24, 144, 288);
>> MoveLineBy("LS1", 18, 36);
>> MoveStart("BM1", 0, 12);
>> var location = new Location(-144, -360);
>> MoveEnd("LS1", location);
```

## 6. Apply Loads

- a. Rectangular, Circular, Tubular, and Rings loads can be added to the model with various input options (i.e. service case, location, pressure type, etc.). Note: Use a negative magnitude to apply loads in the negative

global direction.

```
Script
>> RectangularLoad(0, 0, 12, 24, -50, 10, 20);
>> CircularLoad("D", 0, 0, 24, -50, 10, 20);
>> TubeLoad(0, 0, 24, 48, 6, -50, -100, 10, 20, true);
>> RingLoad("D", 0, 0, 36, 6, -50, -100, 10, 20, true);
```

- b. Individual boundaries or the full boundary can also be loaded using the command line.

```
Script
>> LoadBoundary("S1", -50, 10, 20);
>> LoadFullBoundary("D", -50, -100, 10, 20, true);
```

- c. A variety of options are available to apply loads to Load Points using the command line as outlined on the [Commands](#) page. In the example below, Load Point LP1 is loaded in the D service case with a 5 kip vertical force in the Z, a 100 kip-in moment about the X-axis, and a 200 kip-in moment about the Y-axis (assuming the current unity style is Kips & Inches).

```
Script
>> PointLoad("D", "LP1", 5, 100, 200)
Loaded LP1
```

- d. Beams are loaded using the LoadBeam command which can take various inputs.

```
Script
>> LoadBeam("BM1", 5, 100, 200, true, true);
>> LoadBeam("L", "BM1", 5, 100, 200, true, false);
```

## 7. Analysis Settings

- a. The command line can be used to set the mesh to a standard value (Coarse, Medium, or Fine) or to a User Defined value.

```
Script
>> SetMeshMedium();
>> SetMeshCustom(6000);
```

- b. The SetMaterial() command can be used to set the material for the boundary and beams in the project. This command launches the material dialog box so that a material can be chosen. Alternatively, the name of the material can be used in the command to directly set the material.

```
Script
>> SetMaterial()
Boundary and beam material changed to IES\Concrete\Concrete (F'c = 4 ksi).
>> SetMaterial("Concrete (F'c = 3 ksi)")
Boundary and beam material changed to Concrete (F'c = 3 ksi).
```

## 8. Obtain Results

- a. The maximum or minimum displacement, shear or moment across all result cases or for a specified result case can be obtained using command line. Also, the command line can output the displacement, moment, shear, or bearing pressure at a specific location for a defined result case.

```
Script
>> Displacement(false)
    -0.156335236483128
>> MomentX("1. D", false)
    -4.02690469152969
>> ShearY("2. 1.2D+1.6L+0.3S", 0, -150)
    0.146922500717636
```

- b. The maximum or minimum Point Support or Line Support force or moment across all result cases can be obtained using command line. Also, the command line can output the force at a specified Point Support or Line Support Point for a defined result case.

```
Script
>> PointSupportFZ("PS1", true)
    80.7006383070048
>> LineSupportMoment("1. 1.4D", "LS1")
    6064.21964525991
```

## 9. Miscellaneous

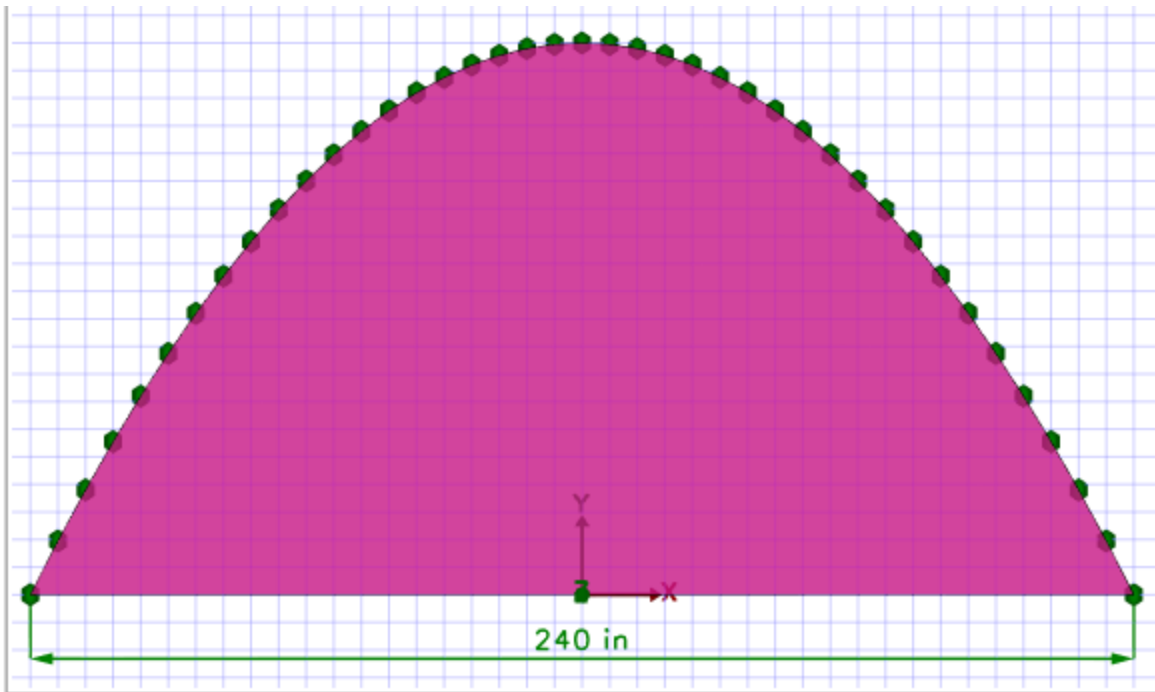
- a. Integer Math: In the command line, the quotient of two integers is an integer which may not produce the intended result. See the [C# Numeric Types Tutorial](#) for more information.

```
Script
>> 1/2
    0
>> 1/2.0
    0.5
```

## C# Programming

The command line accepts any valid command in the [C# Programming Language](#), allowing the use of if statements, for loops, etc. In the example below, a command is defined to describe the y coordinate of a parabola terms of x. The Pow(Double, Double) method of the [.Net System Math Class](#) is used to perform squared operation. A four loop is used to add the x and y coordinates of the boundary to lists which are then used to define the boundary. In addition to using the command line to program directly, more complex [Scripts](#) can be generated in any text file and read into the program.





#### Script

```
>> double y(double x) => 120 - Math.Pow(x, 2)/120.0;
>> var xCoordinates = new List<double>();
>> var yCoordinates = new List<double>();
>> for(double x = -120; x <= 120; x = x + 6) {xCoordinates.Add(x); yCoordinates.Add(y(x));}
>> AddBoundary(xCoordinates, yCoordinates)
S001
```

## 5.2 Commands

### Script Command Categories

- [User Interface](#)
- [General](#)
- [Vertices](#)
- [Boundaries](#)
- [Point Supports](#)
- [Line Supports](#)
- [Beams](#)
- [Move Beam/Line Support](#)
- [Service Cases](#)
- [Load Points](#)
- [Boundary Loads](#)
- [Point Loads](#)
- [Beam Loads](#)
- [Delete Loads](#)
- [Load Combinations](#)
- [Analysis Settings](#)
- [Status](#)
- [Pipeline](#)
- [Result Cases](#)
- [Plate Results](#)

# ConcreteBending 8.0 User's Guide

- [Boundary Results](#)
- [Point Support Results](#)
- [Line Support Results](#)
- [Design Results](#)
- [Report](#)

	Command	Description	Example Input	Example Result
<b>User Interface</b> <a href="#">(BACK TO TOP)</a>	Clear	Clears all text in the command line and clears any stored variables		
	Browse	Launches the Open dialog box to navigate to an external script to run		
	<i>Up/Down Arrows</i>	Use the " <i>Up Arrow</i> " and " <i>Down Arrow</i> " keys to navigate the command line history		
	<i>Esc</i>	Press the " <i>Esc</i> " key while a script is running to end the script run		
<b>General</b> <a href="#">(BACK TO TOP)</a>	Help()	Launches the Help File and navigates to the command line overview page		
	Help(command)	Launches the Help File and navigates to the specified command	Help("AddBoundary")	Launches the Help File and navigates to the AddBoundary command
	SetUnits(style)	Sets the unit style to a default or custom style in the program	SetUnits("Canadian")	Sets the programs unit style to Canadian
	SetDisplayPrecision(DecimalPlaces)	Sets the number of decimal places displayed	SetDisplayPrecision(4)	Sets the precision to 4 decimal places
	Select(names[])	Selects a specified item(s) (e.g. boundary, vertex, etc.) in the model	Select("B1", "V1")	Selects boundary B1 and vertex V1 in the model
	Delete()	Deletes the selected model object(s) and/or load(s)	Select boundary B1, element Bm1, and vertex V1 then enter Delete()	Deletes boundary B1, element Bm1, and vertex V1 in the model
	Delete(names[])	Deletes a specified item(s) (e.g. boundaries, load points, point supports, etc.) in the model	Delete("B1", "LP1", "PS1")	Deletes boundary B1, load point LP1, and point support PS1 in the model
	DeleteAll()	Deletes everything in the model		
	Zoom(name)	Zooms to a specified item (e.g. boundary, load point, point support, etc.) in the	Zoom("V1")	Zooms into vertex V1 in the model

	model		
Print(List<names>)	Prints the list of items in a comma-delimited format	Print(Boundaries())	Prints the list of boundaries in the project in a comma-delimited format
List(List<names>)	Lists the list of items with one item per line	List(Vertices())	Lists the list of vertices with one item per line
PickConcrete()	Opens the Concrete Material Database dialog box	SetMaterial(PickConcrete())	Opens the Concrete Material Database dialog box to set the concrete for the model's boundary and beams
SetTitle()	Sets the Title for the project	SetTitle("Elevated Slab Design")	Sets the Title for the project to "Elevated Slab Design"
SetBillingReference()	Sets the Billing Reference for the project	SetBillingReference("ABC Architects")	Sets the Billing Reference for the project to "ABC Architects"
SetProjectNotes()	Adds a project note	SetProjectNotes("Analysis & design complete.")	Adds note "Analysis & design complete." to the project.
SetNodalTolerance()	Sets the Nodal Tolerance	SetNodalTolerance(0.075)	Sets the Nodal Tolerance to 0.075
<b>Vertices</b> <a href="#">(BACK TO TOP)</a>			
Vertices()	Returns a list of all the vertices in the project	Print(Vertices())	Prints the list of all the vertices in the project in a comma-delimited format
MoveTo(name, X, Y)	Moves a specified vertex to a defined location	MoveTo("V1", 0, 0)	Moves vertex V1 to the origin
MoveTo(name, location)	Moves a specified vertex to a predefined location	var location1 = new Location(10, 20); MoveTo("V1", location1)	Moves vertex V1 to location1
MoveBy(name, distanceX, distanceY)	Moves a specified vertex by a specified distance in each global direction	MoveBy("V1", 5, 10)	Moves vertex V1 by 5 in the global X direction and 10 in the global Y direction
X(name)	Returns the global X-coordinate of the specified vertex	X("V1")	Returns the global X-coordinate of vertex V1
Y(name)	Returns the global Y-coordinate of the specified vertex	Y("V1")	Returns the global Y-coordinate of vertex V1
<b>Boundaries</b>			
Boundaries()	Returns a list of all	Print(Boundaries())	Prints the list of

[\(BACK TO TOP\)](#)

	the boundaries in the project		all the boundaries in the project in a comma-delimited format
AddBoundary(List<X>, List<Y>)	Adds a boundary defined by the a list of X coordinates and list of Y coordinates.	<pre>var X = new List&lt;double&gt;(){0, 1, 0}; var Y = new List&lt;double&gt;(){0, 0, 1}; AddBoundary(X, Y)</pre>	Adds a boundary defined by coordinate lists X and Y
AddBoundary(Location[])	Adds a boundary defined by and array of locations	<pre>var location1 = new Location(0, 0); var location2 = new Location(1, 0); var location3 = new Location(0, 1); AddBoundary(location1, location2, location3)</pre>	Adds a boundary defined by location1, location2, and location3
AddRectangleBoundary(centerX, centerY, width, height)	Adds a rectangular boundary of specified width and height at a defined location	AddRectangleBoundary(1, 2, 5, 15)	Adds a 5 wide by 10 tall rectangular boundary centered at {1, 2}
AddCircleBoundary(centerX, centerY, radius)	Adds a circular boundary of specified radius at a defined location	AddCircleBoundary(1, 2, 5)	Adds a circular boundary of radius 5 centered at {1, 2}
AddPolygonBoundary(centerX, centerY, radius, numberSides, angle)	Adds a polygon boundary of specified radius at a defined location	AddPolygonBoundary(1, 2, 5, 6, 45)	Adds a 6 sided polygon boundary of radius 5 centered at {1, 2} with a rotation angle of 45
ToggleHole(boundary)	Toggles the Hole? parameter for the specified boundary on or off	ToggleHole("B1")	Toggles the Hole? parameter on or off for boundary B1
Thickness(boundary, thickness)	Sets the thickness to a specified value for the defined boundary	Thickness("B1", 12)	Sets the thickness for boundary B1 to 12
ToggleSelfWeight(boundary)	Toggles the Add Self Weight parameter for the specified boundary on or off	ToggleSelfWeight("B1")	Toggles the Add Self Weight parameter on or off for boundary B1
Height(boundary, height)	Modifies the height of a rectangular or circular boundary	Height("B1", 120)	Changes the height of boundary B1 to 120
Width(boundary, width)	Modifies the width of a rectangular or circular boundary	Width("B1", 120)	Changes the width of boundary B1 to 120
Theta(boundary, theta)	Modifies the theta of a rectangular or polygon boundary	Theta("B1", 45)	Changes theta of boundary B1 to 45
Radius(boundary, radius)	Modifies the radius of a circular or polygon boundary	Radius("B1", 120)	Changes the radius of boundary B1 to 120
SideCount(boundary, sideCount)	Modifies the number of sides for a	SideCount("B1", 6)	Changes the number of sides

	polygon boundary		for boundary B1 to 6
SideLength(boundary, length)	Modifies the side length for a polygon boundary	SideLength("B1", 120)	Changes the side length for boundary B1 to 120
MoveCenterTo(boundary, centerX, centerY)	Moves the center of the a circular, rectangular, or polygon boundary to a defined location	MoveCenterTo("B1", 120, 240)	Moves the center of boundary B1 to {120, 240}
MoveCenterTo(boundary, location)	Moves the center of the a circular, rectangular, or polygon boundary to a defined location	var location1 = new Location(120, 240); MoveCenterTo("B1", location1)	Moves the center of boundary B1 to location1
MoveCenterBy(boundary, distanceX, distanceY)	Moves the center of the a circular, rectangular, or polygon slab by a specified amount	MoveCenterBy("F1", 120, 240)	Moves the center of slab F1 by 120 in the X-direction and 240 in the Y-direction
SingleMat_Optimize(boundary, xBarsTop, bottomCover, (string, double)[] barPatterns)	Sets the design approach for the boundary to optimize, defines the reinforcement details, and sets the reinforcement search patterns for the single mat	SingleMat_Optimize("B1", true, 3, ("#4", 18), ("#4", 12), ("#5", 18), ("#5", 12))	Sets the design approach for boundary B1 to optimize, defines the rebar directions and cover, and sets the search patterns as #4 @ 18, #4 @ 12, #5 @ 18, #4 @ 12, for the single mat
DoubleMat_Optimize(boundary, topXTop, topCover, bottomXTop, bottomCover, (string, double)[] barPatterns)	Sets the design approach for the boundary to optimize, defines the reinforcement details, and sets the reinforcement search patterns for both mats	DoubleMat_Optimize("B1", true, 2, true, 3, ("#4", 18), ("#4", 12), ("#5", 18), ("#5", 12))	Sets the design approach for boundary B1 to optimize, defines the rebar directions and covers, and sets the search patterns as #4 @ 18, #4 @ 12, #5 @ 18, #4 @ 12, for both mats
SingleMat_Specify(boundary, xBarsTop, bottomCover, xSize, xSpacing, ySize, ySpacing)	Sets the design approach for the boundary to specify and defines the reinforcement mat and the reinforcement details	SingleMat_Specify("B1", true, 2, "#6", 12, "#8", 16)	Sets the design approach for boundary B1 to specify and defines the single mat of reinforcement
DoubleMat_Specify(boundary, topXTop, topCover, bottomXTop, bottomCover, topXSize, topXSpacing, topYSize, topYSpacing, bottomXSize, bottomXSpacing, bottomYSize, bottomYSpacing)	Sets the design approach for the boundary to specify and defines the reinforcement mats (top and bottom) and the reinforcement details	DoubleMat_Specify("B1", true, 2, true, 3, "#6", 12, "#6", 12, "#8", 16, "#8", 16)	Sets the design approach for boundary B1 to specify and defines the two mats of reinforcement
PointSupports()	Returns a list of all	Print(PointSupports())	Prints the list of

## Supports

[\(BACK TO TOP\)](#)

	the point supports in the project		all the point supports in the project in a comma-delimited format
AddPointSupport(X, Y)	Adds a default point support at a specified coordinate	AddPointSupport(0, 0)	Adds a default point support at the origin
AddPointSupport(location)	Adds a default point support at a specified location	var location1 = new Location(12, 24); AddPointSupport(location1)	Adds a default point support at location1
NamePointSupport(currentName, newName)	Renames a point support	NamePointSupport("PS1", "PS2")	Changes the name of point support PS1 to PS2
SquarePointSupportSize(support, size)	Sets the specified point support shape to square and defines its size	SquarePointSupportSize("PS1", 8)	Sets point support PS1 as a square and defines its size as 8
RectanglePointSupportSize(support, sizeX, sizeY)	Sets the specified point support shape to rectangular and defines its size	RectanglePointSupportSize("PS1", 6, 12)	Sets point support PS1 as a rectangle and defines its size as 6x12
CirclePointSupportSize(support, radius)	Sets the specified point support shape to circle and defines its size	CirclePointSupportSize("PS1", 6)	Sets point support PS1 as a circle and defines its radius as 6
IShapePointSupportSize(support, sizeX, sizeY)	Sets the specified point support shape to an I-shape and defines its size	IShapePointSupportSize("PS1", 9, 18)	Sets point support PS1 as an I-shape and defines its size as 9 in the x-direction and 18 in the y-direction
FixPointSupport(support, dof)	Sets the specified degree of freedom to fixed for the defined point support	FixPointSupport("PS1", "DZ")	Sets Force-Z to fixed for point support PS1
FreePointSupport(support, dof)	Sets the specified degree of freedom to free for the defined point support	FreePointSupport("PS1", "RX")	Sets Moment-X to free for point support PS1
CustomPointSupport(support, dof, stiffness)	Sets the specified degree of freedom to a spring support with specified stiffness for the defined point support	CustomPointSupport("PS1", "RY", 100)	Sets Moment-Y to Specified-K for point support PS1 with a Stiffness-Ry = 100
RefinePointSupport(support, offset, startLength, endLength)	Sets the Refine parameter for the specified point support to true and defines the mesh refinement parameters	RefinePointSupport("PS1", 6, 0.5, 4)	Sets the Refine parameter for point support PS1 to true and set the offset to 6, the element length at start to 0.5, and the element length

	UnRefinePointSupport(support)	Sets the Refine parameter for the specified point support to false	UnRefinePointSupport("PS1")	at end to 4 Sets the Refine parameter for point support PS1 to false
	MovePointSupportTo(support, X, Y)	Moves specified point support to defined coordinates	MovePointSupportTo("PS1", 12, 24)	Moves point support PS1 to {12, 24}
	MovePointSupportTo(support, location)	Moves specified point support to defined location	var location1 = new Location(12, 24); MovePointSupportTo("PS1", location1)	Moves point support PS1 to location1
	MovePointSupportBy(support, distanceX, distanceY)	Moves specified point support by a specified distance in each global direction	MovePointSupportBy("PS1", 12, 24)	Moves point support PS1 by 12 in the X-direction and 24 in the Y-direction
<b>Line Supports</b> <a href="#">(BACK TO TOP)</a>	LineSupports()	Returns a list of all the line supports in the project	Print(LineSupports())	Prints the list of all the line supports in the project in a comma-delimited format
	AddLineSupport(startX, startY, endX, endY)	Adds a line support between start coordinates and end coordinates	AddLineSupport(0, 0, 120, 240)	Adds a line support between {0, 0} and {120, 240}
	AddLineSupport(startLocation, endLocation)	Adds a line support between a start location and an end location	var location1 = new Location(0, 0); var location2 = new Location(120, 240); AddLineSupport(location1, location2)	Adds a line support between location1 and location2
	AddLineSupport(name, startX, startY, endX, endY, thickness)	Adds a line support between start coordinates and end coordinates with a defined name and thickness	AddLineSupport("LS1", 0, 0, 120, 240, 12)	Adds a line support named LS1 between {0, 0} and {120, 240} that is 12 thick
	AddLineSupport(name, startLocation, endLocation, thickness)	Adds a line support between a start location and an end location with a defined name and thickness	var location1 = new Location(0, 0); var location2 = new Location(120, 240); AddLineSupport("LS1", location1, location2, 12)	Adds a line support named LS1 between location1 and location2 that is 12 thick
	FixLineSupport(support)	Sets the moment fixity at the support to fixed	FixLineSupport("LS1")	Sets the moment fixity for line support LS1 to fixed
	PinLineSupport(support)	Sets the moment fixity at the support to pinned	PinLineSupport("LS1")	Sets the moment fixity for line support LS1 to pinned
	CustomLineSupport(support, stiffness)	Sets the moment fixity at the support to a spring with a specified stiffness	CustomLineSupport("LS1", 100)	Sets the moment fixity for line support LS1 to a spring of stiffness 100
<b>Beams</b> <a href="#">(BACK TO TOP)</a>	Beams()	Returns a list of all the beams in the project	Print(Beams())	Prints the list of all the beams in the project in a comma-delimited format

## Move Beam/Line Support

[\(BACK TO TOP\)](#)

AddBeam(startX, startY, endX, endY)	Adds a beam between start coordinates and end coordinates	AddBeam(0, 0, 120, 240)	Adds a beam between {0, 0} and {120, 240}
AddBeam(startLocation, endLocation)	Adds a beam between a start location and an end location	var location1 = new Location(0, 0); var location2 = new Location(120, 240); AddBeam(location1, location2)	Adds a beam between location1 and location2
AddBeam(name, startX, startY, endX, endY, depth, width, includeSelfWeight)	Adds a beam between start coordinates and end coordinates with a defined name, depth, , and width and with the self weight included or excluded	AddBeam("BM1", 0, 0, 120, 240, 24, 12, true)	Adds a beam named BM1 between {0, 0} and {120, 240} that is 24 deep, 12 wide, and includes its self weight
AddBeam(name, startLocation, endLocation, depth, width, includeSelfWeight)	Adds a beam between a start location and an end location with a defined name, depth, and width and with the self weight included or excluded	var location1 = new Location(0, 0); var location2 = new Location(120, 240); AddBeam("BM1", location1, location2, 24, 12, true)	Adds a beam named BM1 between location1 and location2 that is 24 deep, 12 wide, and includes its self weight
SetBeamTopFlush(beam)	Sets the vertical offset for the specified beam to Top Flush	SetBeamTopFlush("BM1")	Sets the vertical offset for beam BM1 to Top Flush
SetBeamCentered(beam)	Sets the vertical offset for the specified beam to Centered	SetBeamCentered("BM1")	Sets the vertical offset for beam BM1 to Centered
SetBeamBottomFlush(beam)	Sets the vertical offset for the specified beam to Bottom Flush	SetBeamBottomFlush("BM1")	Sets the vertical offset for beam BM1 to Bottom Flush
MoveLineTo(beamOrLineSupport, startX, startY, endX, endY)	Moves specified beam or line support to a defined start coordinate and end coordinate	MoveLineTo("L1", 0, 0, 120, 240)	Moves beam or line support L1 to span between coordinates {0, 0} and {120, 240}
MoveLineTo(beamOrLineSupport, startLocation, endLocation)	Moves specified beam or line support to a defined start location and end location	var location1 = new Location(0, 0); var location2 = new Location(120, 240); MoveLineTo("L1", location1, location2)	Moves beam or line support L1 to span between location1 and locaiton2
MoveLineBy(beamOrLineSupport, distanceX, distanceY)	Moves specified beam or line support by specified amounts in the X-direction and the Y-direction	MoveLineBy("L1", 12, 24)	Moves beam or line support L1 by 12 in the X-direction and 24 in the Y-direction
MoveStart(beamOrLineSupport, X, Y)	Moves the start point of a specified beam or line support to a defined coordinate	MoveStart("L1", 6, 12)	Moves the start coordinate of beam or line support L1 to {6, 12}
MoveStart(beamOrLineSupport,	Moves the start point	var location1 = new Location(0, 0);	Moves the start



	startLocation)	of a specified beam or line support to a defined location	MoveStart("L1", location1)	coordinate of beam or line support L1 to location1
	MoveEnd(beamOrLineSupport, X, Y)	Moves the end point of a specified beam or line support to a defined coordinate	MoveEnd("L1", 6, 12)	Moves the end coordinate of beam or line support L1 to {6, 12}
	MoveEnd(beamOrLineSupport, endLocation)	Moves the end point of a specified beam or line support to a defined location	var location1 = new Location(120, 120); MoveEnd("L1", location1)	Moves the end coordinate of beam or line support L1 to location1
<b>Service Cases</b> <a href="#">(BACK TO TOP)</a>	ServiceCases()	Returns a list of all the service cases in the project	Print(ServiceCases())	Prints the list of all the service cases in the project in a comma-delimited format
	AddServiceCase(name, source)	Adds a service case with a specified name and source	AddServiceCase("S-Unbalanced", "Snow")	Add service case S-Unbalanced with a Snow load source
	AddServiceCase(name, source, includeInCombos, pattern)	Adds a service case with a specified name and source and defines the pattern ID and if the case is included in the building code combinations	AddServiceCase("S-Unbalanced", "Snow", true, 1)	Add service case S-Unbalanced with a Snow load source and includes the service case in the building code combination and sets the pattern ID to 1
<b>Load Points</b> <a href="#">(BACK TO TOP)</a>	LoadPoints()	Returns a list of all the load points in the project	Print(LoadPoints())	Prints the list of all the load points in the project in a comma-delimited format
	AddLoadPoint(X, Y)	Adds a default load point at a specified coordinate	AddLoadPoint(0, 0)	Adds a default load point at the origin
	AddLoadPoint(location)	Adds a default load point at a specified location	var location1 = new Location(12, 24); AddLoadPoint(location1)	Adds a default load point at location1
	NameLoadPoint(currentName, newName)	Renames a load point	NameLoadPoint("LP1", "LP2")	Changes the name of point support PS1 to PS2
	SquareLoadPointSize(point, size)	Sets the specified load point shape to square and defines its size	SquareLoadPointSize("LP1", 8)	Sets load point LP1 as a square and defines its size as 8
	RectangleLoadPointSize(point, sizeX, sizeY)	Sets the specified load point shape to rectangular and defines its size	RectangleLoadPointSize("LP1", 6, 12)	Sets load point LP1 as a rectangle and defines its size

## Boundary Loads

[\(BACK TO TOP\)](#)

CircleLoadPointSize(point, radius)	Sets the specified load point shape to circle and defines its size	CircleLoadPointSize("LP1", 6)	as 6x12 Sets load point LP1 as a circle and defines its radius as 6
OctagonLoadPointSize(point, radius)	Sets the specified load point shape to an octagon and defines its size	OctagonLoadPointSize("LP1", 9)	Sets load point LP1 as an octagon and defines its radius as 9
SetloadPointTheta(point, theta)	Sets the rotation for a specified load point	SetLoadPointTheta("LP1", 45)	Sets the rotation for load point LP1 to 45
RefineLoadPoint(point, offset, startLength, endLength)	Sets the Refine parameter for the specified load point to true and defines the mesh refinement parameters	RefineLoadPoint("LP1", 6, 0.5, 4)	Sets the Refine parameter for load point LP1 to true and set the offset to 6, the element length at start to 0.5, and the element length at end to 4
UnRefineLoadPoint(support)	Sets the Refine parameter for the specified load point to false	UnRefineLoadPoint("LP1")	Sets the Refine parameter for load point LP1 to false
MoveLoadPointTo(point, X, Y)	Moves specified load point to defined coordinates	MoveLoadPointTo("LP1", 12, 24)	Moves load point LP1 to {12, 24}
MoveLoadPointTo(point, location)	Moves specified load point to defined location	var location1 = new Location(12, 24); MoveLoadPointTo("LP1", location1)	Moves load point LP1 to location1
MoveLoadPointBy(point, distanceX, distanceY)	Moves specified load point by a specified distance in each global direction	MoveLoadPointBy("LP1", 12, 24)	Moves load point LP1 by 12 in the X-direction and 24 in the Y-direction
RectangularLoad(X, Y, widthX, widthY, UniformPressure, MX, MY)	Applies a widthX by widthY rectangular load centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	RectangularLoad(0, 0, 2, 4, -50, 10, 20)	Applies a 2 by 4 rectangular load centered at {0, 0} with a specified uniform pressure of -50 with 10 and 20 overturning loads in the current service case
RectangularLoad(ServiceCase, X, Y, widthX, widthY, UniformPressure, MX, MY)	Applies a widthX by widthY rectangular load centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case	RectangularLoad("D", 0, 0, 2, 4, -50, 10, 20)	Applies a 2 by 4 rectangular load centered at {0, 0} with a specified uniform pressure of -50 with 10 and 20 overturning loads in the "D" service case
RectangularLoad(X, Y, widthX, widthY, startPressure, endPressure,	Applies a widthX by widthY rectangular	RectangularLoad(0, 0, 2, 4, -50, -100, 10, 20, true)	Applies a 2 by 4 rectangular load

MX, MY, linearX)	load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case		centered at {0, 0} with a linear pressure varying from -50 to -100 in the X-direction with 10 and 20 overturning loads in the current service case
RectangularLoad(serviceCase, X, Y, widthX, widthY, startPressure, endPressure, MX, MY, linearX)	Applies a widthX by widthY rectangular load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	RectangularLoad("D", 0, 0, 2, 4, -50, -100, 10, 20, true)	Applies a 2 by 4 rectangular load centered at {0, 0} with a linear pressure varying from -50 to -100 in the X-direction with 10 and 20 overturning loads in the "D" service case
CircularLoad(X, Y, radius, uniformPressure, MX, MY)	Applies a circular load with a specified radius centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	CircularLoad(0, 0, 4, -50, 10, 20)	Applies a circular load with a 4 radius centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the current service case
CircularLoad(serviceCase, X, Y, radius, uniformPressure, MX, MY)	Applies a circular load with a specified radius centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case	CircularLoad("D", 0, 0, 4, -50, 10, 20)	Applies a circular load with a 4 radius centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the "D" service case
CircularLoad(X, Y, radius, startPressure, endPressure, MX, MY, linearX)	Applies a circular load with a specified radius centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case	CircularLoad(0, 0, 4, -50, -100, 10, 20, true)	Applies a circular load with a 4 radius centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the current service case
CircularLoad(serviceCase, X, Y, radius, startPressure, endPressure, MX, MY, linearX)	Applies a circular load with a specified radius centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	CircularLoad("D", 0, 0, 4, -50, -100, 10, 20, true)	Applies a circular load with a 4 radius centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20

	case		overturning loads in the "D" service case
TubeLoad(X, Y, widthX, widthY, thickness, uniformPressure, MX, MY)	Applies a widthX by widthY tube load centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	TubeLoad(0, 0, 2, 4, 0.5, -50, 10, 20)	Applies a 2 by 4 tube load centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the current service case
TubeLoad(serviceCase, X, Y, widthX, widthY, thickness, uniformPressure, MX, MY)	Applies a widthX by widthY tube load centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case	TubeLoad("D", 0, 0, 2, 4, 0.5, -50, 10, 20)	Applies a 2 by 4 tube load centered at {0, 0} with a -50 uniform pressure in the X-direction with 10 and 20 overturning loads in the "D" service case
TubeLoad(X, Y, widthX, widthY, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a widthX by widthY tube load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case	TubeLoad(0, 0, 2, 4, 0.5, -50, -100, 10, 20, true)	Applies a 2 by 4 tube load centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the current service case
TubeLoad(serviceCase, X, Y, distanceX, distanceY, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a widthX by widthY tube load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	TubeLoad("D", 0, 0, 2, 4, 0.5, -50, -100, 10, 20, true)	Applies a 2 by 4 tube load centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the "D" service case
RingLoad(X, Y, radius, thickness, uniformPressure, MX, MY)	Applies a ring load with a specified radius and thickness centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	RingLoad(0, 0, 4, 1, -50, 10, 20)	Applies a ring load with a 4 radius and 1 thickness centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the current service case
RingLoad(serviceCase, X, Y, radius, thickness, uniformPressure, MX, MY)	Applies a ring load with a specified radius and thickness centered at {X, Y} with a specified	RingLoad("D", 0, 0, 4, 1, -50, 10, 20)	Applies a ring load with a 4 radius and 1 thickness centered at {0, 0}

	uniform pressure with MX and MY overturning loads in the defined service case		with a -50 uniform pressure with 10 and 20 overturning loads in the "D" service case
RingLoad(X, Y, radius, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a ring load with a specified radius and thickness centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case	RingLoad(0, 0, 4, 1, -50, -100, 10, 20, true)	Applies a ring load with a 4 radius and 1 thickness centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the current service case
RingLoad(serviceCase, X, Y, radius, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a ring load with a specified radius and thickness centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	RingLoad("D", 0, 0, 4, 1, -50, -100, 10, 20, true)	Applies a ring load with a 4 radius and 1 thickness centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the "D" service case
LoadBoundary(uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the selected boundary in the current service case	LoadBoundary(-50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to the selected boundary in the current service case
LoadBoundary(slab, uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the specified boundary in the current service case	LoadBoundary("S1", -50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to boundary S1 in the current service case
LoadBoundary(serviceCase, boundary, uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the specified boundary in the defined service case	LoadBoundary("D", "S1", -50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to boundary S1 in the "D" service case
LoadFullBoundary(uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the full boundary in the current service case	LoadFullBoundary(-50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to the full boundary in the

	<code>LoadFullBoundary(serviceCase, uniformPressure, MX, MY)</code>	Applies a uniform pressure with MX and MY overturning loads to the full boundary in the defined service case	<code>LoadFullBoundary("D", -50, 10, 20)</code>	current service case Applies a -50 uniform pressure with 10 and 20 overturning loads to the full boundary in the "D" service case
	<code>LoadFullBoundary(startPressure, endPressure, MX, MY, linearX)</code>	Applies a linear pressure with MX and MY overturning loads to the full boundary in the current service case	<code>LoadFullBoundary(-50, -100, 10, 20, true)</code>	Applies a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads to the full boundary in the current service case
	<code>LoadFullBoundary(serviceCase, startPressure, endPressure, MX, MY, linearX)</code>	Applies a linear pressure with MX and MY overturning loads to the full boundary in the defined service case	<code>LoadFullBoundary("D", -50, -100, 10, 20, true)</code>	Applies a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads to the full boundary in the "D" service case
<b>Point Loads</b> <a href="#">(BACK TO TOP)</a>	<code>PointLoad(FZ, MX, MY)</code>	Applies specified force and moments to the selected load point in the current service case	<code>PointLoad(5, 100, 200)</code>	Applies force and moments to the selected load point in the current service case
	<code>PointLoad(loadPoint, FZ, MX, MY)</code>	Applies specified force and moments to the defined load point in the current service case	<code>PointLoad("LP1", 5, 100, 200)</code>	Applies forces and moments to load point "LP1" in the current service case
	<code>PointLoad(serviceCase, loadPoint, FZ, MX, MY)</code>	Applies specified force and moments to the defined load point in the specified service case	<code>PointLoad("D", "LP1", 5, 100, 200)</code>	Applies forces and moments to load point "LP1" in the D service case
<b>Beam Loads</b> <a href="#">(BACK TO TOP)</a>	<code>LoadBeam(FZ, MX, MY, distributed = false, local = false)</code>	Applies specified force and moments to the selected beam in the current service case as a resultant or distributed load in the global or local direction	<code>LoadBeam(5, 100, 200)</code>	Applies force and moments to the selected beam in the current service case. The loads are applied as resultants in the global direction since "distributed" and "local" are false by default.
	<code>LoadBeam(beam, FZ, MX, MY, distributed = false, local = false)</code>	Applies specified force and moments to the defined beam in the current service case as a resultant or distributed load in	<code>LoadBeam("BM1", 5, 100, 200, true, true)</code>	Applies force and moments to beam BM1 in the current service case. The loads are

		the global or local direction		applied as distributed loads in the local direction.
	LoadBeam(serviceCase, beam, FZ, MX, MY, distributed = false, local = false)	Applies specified forces and moments to the defined beam in the defined service case as a resultant or distributed load in the global or local direction	LoadBeam("D", "BM1", 5, 100, 200, true, false)	Applies force and moments in the D service case to the beam BM1 in the current service case. The loads are applied as distributed loads in the global direction.
<b>Delete Loads</b> <a href="#">(BACK TO TOP)</a>	ClearLoads()	Deletes all of the loads in the active service case		
	ClearLoads(case)	Deletes all of the loads in the specified service case	ClearLoads("L")	Deletes all of the loads in the L service case
	DeleteLoad(object)	Deletes the loads on a specified object in the active service case	DeleteLoad("B1")	Deletes the beam loads on beam B1 in the active service case
	DeleteLoad(case, object)	Deletes the loads on a specified object in the specified service case	DeleteLoad("L", "B1")	Deletes the beam loads on beam B1 in the L service case
<b>Load Combinations</b> <a href="#">(BACK TO TOP)</a>	AddCustomCombo(name, comboType, (factor, case[]))	Adds a custom load combination to the project	AddCustomCombo("D + 0.75L", "Allowable (ASD)", (1.0, "D"), (0.75, "L"))	Adds custom D + 0.75L Allowable (ASD) load combination
<b>Analysis Settings</b> <a href="#">(BACK TO TOP)</a>	SetMeshCoarse()	Sets the Mesh Refinement to Coarse		
	SetMeshMedium()	Sets the Mesh Refinement to Medium		
	SetMeshFine()	Sets the Mesh Refinement to Fine		
	SetMeshCustom(elementCount)	Sets the Mesh Refinement to User Defined and sets the Element Count to a specified value	SetMeshCustom(6000)	Sets the Mesh Refinement to User Defined and sets the Element Count to 6000
	SetEFactor(factor)	Sets the Concrete Elastic Modulus Factor in the project	SetEFactor(2)	Sets the Concrete Elastic Modulus Factor to 2 in the project
	ThicknessOverlap(setting)	Changes the Thickness Overlap setting to the specified value	ThicknessOverlap("Smallest")	Changes the Thickness Overlap setting to Smallest Thickness
	SetMaterial()	Opens the Material		

# ConcreteBending 8.0 User's Guide

		Database dialog box to select the concrete material that is used for the slabs and beams in the project		
	SetMaterial(concrete)	Sets the concrete material that is used for the slabs and beams in the project	SetMaterial("Concrete (F'c = 3.5 ksi)")	Sets the concrete material that is used for the slabs and beams in the project to Concrete (F'c = 3.5 ksi)
<b>Status</b> <a href="#">(BACK TO TOP)</a>	Status(itemTitle)	Returns the Project Status of the specified item.  -1 = Failing  0 = Not found or not available  1 = Has a warning  2 = Ok	Status("Slab Shear")	Returns the Project Status of the Slab Shear
<b>Pipeline</b> <a href="#">(BACK TO TOP)</a>	Meshing()	Pauses the script until the meshing completes. Note: Only used for external scripts and must be preceded by "await"	await Meshing();	Pauses the external script until the meshing completes
	Analysis()	Pauses the script until the analysis completes. Note: Only used for external scripts and must be preceded by "await"	await Analysis();	Pauses the external script until the analysis completes
	Design()	Pauses the script until the design completes. Note: Only used for external scripts and must be preceded by "await"	await Design();	Pauses the external script until the design completes
<b>Result Cases</b> <a href="#">(BACK TO TOP)</a>	Results()	Returns a list of all the result cases in the project	Print(Results())	Prints the list of all the result cases in the project in a comma-delimited format
	ServiceResults()	Returns a list of all the service result cases in the project	Print(ServiceResults())	Prints the list of all the service result cases in the project in a comma-delimited format
	StrengthResults()	Returns a list of all the strength result cases in the project	Print(StrengthResults())	Prints the list of all the strength result cases in the project in a comma-delimited format
	NoDesignResults()	Returns a list of all	Print(NoDesignResults())	Prints the list of



## Plate Results

[\(BACK TO TOP\)](#)

		the no design result cases in the project		all the no design result cases in the project in a comma-delimited format
	SetVisibleRC(resultCase)	Sets the visible result case in the analysis results	SetVisibleRC("2. D+L")	Sets the visible result case in the analysis results to 2. D+L
	Displacement(max)	Returns the maximum or minimum displacement in the global Z-direction across all result cases	Displacement(false)	Returns the minimum displacement in the global Z-direction across all result cases
	Displacement(result, max)	Returns the maximum or minimum displacement in the global Z-direction for the defined result case	Displacement("1. D", true)	Returns the maximum displacement in the in the global Z-direction for the "1. D" result case
	Displacement(result, X, Y)	Returns the displacement in the global Z-direction at a specified location for the defined result case	Displacement("2. D+L", 6, 12)	Returns the displacement in the global Z-direction at {6, 12} for the "2. D+L" result case
	ShearX(max)	Returns the maximum or minimum shear force that acts on the global X-face of the plate elements (VX) across all result cases	ShearX(false)	Returns the minimum shear force that acts on the global X-face of the plate elements (VX) across all result cases
	ShearX(result, max)	Returns the maximum or minimum shear force that acts on the global X-face of the plate elements (VX) for the defined result case	ShearX("1. D", true)	Returns the maximum shear force that acts on the global X-face of the plate elements (VX) for the defined result case
	ShearX(result, X, Y)	Returns the shear force that acts on the global X-face of the plate elements (VX) at a specified location for the defined result case	ShearX("2. D+L", 6, 12)	Returns the shear force that acts on the global X-face of the plate elements (VX) at {6, 12} for the "2. D+L" result case
	ShearY(max)	Returns the maximum or minimum shear force that acts on the global Y-face of the plate elements (VY) across all result cases	ShearY(true)	Returns the maximum shear force that acts on the global Y-face of the plate elements (VY) across all result cases
	ShearY(result, max)	Returns the maximum or minimum shear force	ShearY("1. D", false)	Returns the minimum shear force that acts

ShearY(result, X, Y)	<p>that acts on the global Y-face of the plate elements (VY) for the defined result case</p> <p>Returns the shear force that acts on the global Y-face of the plate elements (VY) at a specified location for the defined result case</p>	ShearY("2. D+L", 6, 12)	<p>on the global Y-face of the plate elements (VY) for the "1. D" result case</p> <p>Returns the shear force that acts on the global Y-face of the plate elements (VY) at {6, 12} for the "2. D+L" result case</p>
MomentX(max)	<p>Returns the maximum or minimum bending moment that acts on the global X-face of the plate elements (MX) across all result cases</p>	MomentX(true)	<p>Returns the maximum bending moment that acts on the global X-face of the plate elements (MX) across all result cases</p>
MomentX(result, max)	<p>Returns the maximum or minimum bending moment that acts on the global X-face of the plate elements (MX) for the defined result case</p>	MomentX("1. D", false)	<p>Returns the minimum bending moment that acts on the global X-face of the plate elements (MX) for the "1. D" result case</p>
MomentX(result, X, Y)	<p>Returns the bending moment that acts on the global X-face of the plate elements (MX) at a specified location for the defined result case</p>	MomentX("2. D+L", 6, 12)	<p>Returns the bending moment that acts on the global X-face of the plate elements (MX) at {6, 12} for the "2. D+L" result case</p>
MomentY(max)	<p>Returns the maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases</p>	MomentY(false)	<p>Returns the minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases</p>
MomentY(result, max)	<p>Returns the maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) for the defined result case</p>	MomentY("1. D", true)	<p>Returns the maximum bending moment that acts on the global Y-face of the plate elements (MY) for the "1. D" result case</p>
MomentY(result, X, Y)	<p>Returns the bending moment that acts on</p>	MomentY("2. D+L", 6, 12)	<p>Returns the bending</p>

## Boundary Results

[\(BACK TO TOP\)](#)

	the global Y-face of the plate elements (MY) at a specified location for the defined result case		moment that acts on the global Y-face of the plate elements (MY) at a {6, 12} for the "2. D+L" result case
MomentXY(max)	Returns the maximum or minimum twisting moment (MXY) across all result cases	MomentXY(true)	Returns the maximum twisting moment (MXY) across all result cases
MomentXY(result, max)	Returns the maximum or minimum twisting moment (MXY) for the defined result case	MomentXY("1. D", false)	Returns the minimum twisting moment (MXY) for the "1. D" result case
MomentXY(result, X, Y)	Returns the twisting moment (MXY) at a specified location for the defined result case	MomentXY("2. D+L", 6, 12)	Returns the twisting moment (MXY) at a {6, 12} for the "2. D+L" result case
BoundaryDisplacement(boundary, max)	Returns the maximum or minimum displacement in the global Z-direction across all result cases for the specified Boundary	BoundaryDisplacement("B1", false)	Returns the minimum displacement in the global Z-direction across all result cases for boundary B1
BoundaryDisplacement(boundary, result, max)	Returns the maximum or minimum displacement in the global Z-direction for the defined result case for the specified boundary	BoundaryDisplacement("B1", "1. D", true)	Returns the maximum displacement in the in the global Z-direction for the "1. D" result case for boundary B1
BoundaryShearX(boundary, max)	Returns the maximum or minimum shear force that acts on the global X-face of the plate elements (VX) across all result cases for the specified boundary	BoundaryShearX("B1", false)	Returns the minimum shear force that acts on the global X-face of the plate elements (VX) across all result cases for boundary B1
BoundaryShearX(boundary, result, max)	Returns the maximum or minimum shear force that acts on the global X-face of the plate elements (VX) for the defined result case for the specified boundary	BoundaryShearX("B1", "1. D", true)	Returns the maximum shear force that acts on the global X-face of the plate elements (VX) for the defined result case for boundary B1
BoundaryShearY(boundary, max)	Returns the maximum or minimum shear force that acts on the global Y-face of the	BoundaryShearY("B1", true)	Returns the maximum shear force that acts on the global Y-face of the plate

BoundaryShearY(boundary, result, max)	<p>plate elements (VY) across all result cases for the specified boundary</p> <p>Returns the maximum or minimum shear force that acts on the global Y-face of the plate elements (VY) for the defined result case for the specified boundary</p>	BoundaryShearY("B1", "1. D", false)	<p>elements (VY) across all result cases for boundary B1</p> <p>Returns the minimum shear force that acts on the global Y-face of the plate elements (VY) for the "1. D" result case for boundary B1</p>
BoundaryMomentX(boundary, max)	<p>Returns the maximum or minimum bending moment that acts on the global X-face of the plate elements (MX) across all result cases for the specified boundary</p>	BoundaryMomentX("B1", true)	<p>Returns the maximum bending moment that acts on the global X-face of the plate elements (MX) across all result cases for boundary B1</p>
BoundaryMomentX(boundary, result, max)	<p>Returns the maximum or minimum bending moment that acts on the global X-face of the plate elements (MX) for the defined result case for the specified boundary</p>	BoundaryMomentX("B1", "1. D", false)	<p>Returns the minimum bending moment that acts on the global X-face of the plate elements (MX) for the "1. D" result case for boundary b1</p>
BoundaryMomentY(boundary, max)	<p>Returns the maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases for the specified boundary</p>	BoundaryMomentY("B1", false)	<p>Returns the minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases for boundary B1</p>
BoundaryMomentY(boundary, result, max)	<p>Returns the maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) for the defined result case for the specified boundary</p>	BoundaryMomentY("B1", "1. D", true)	<p>Returns the maximum bending moment that acts on the global Y-face of the plate elements (MY) for the "1. D" result case for boundary B1</p>
BoundaryMomentXY(boundary, max)	<p>Returns the maximum or minimum twisting moment (MXY) across all result cases for the specified boundary</p>	BoundaryMomentXY("B1", true)	<p>Returns the maximum twisting moment (MXY) across all result cases for boundary B1</p>

<b>Point Support Results</b> <a href="#">(BACK TO TOP)</a>	BoundaryMomentXY(boundary, result, max)	Returns the maximum or minimum twisting moment (MXY) for the defined result case for the specified boundary	BoundaryMomentXY("B1", "1. D", false)	Returns the minimum twisting moment (MXY) for the "1. D" result case for boundary B1
	PointSupportFZ(result, support)	Returns Force Z for the specified support for the specified result case	PointSupportFZ("1. 1.4D", "PS1")	Returns Force Z for point support PS1 for result case 1. 1.4D
	PointSupportFZ(support, wantMax)	Returns the maximum or minimum Force Z for the specified point support	PointSupportFZ("PS1", true)	Returns the maximum Force Z for point support PS1 across all result cases
	PointSupportMX(result, support)	Returns Moment X for the specified point support for the specified result case	PointSupportMX("1. 1.4D", "PS1")	Returns Moment X for point support PS1 for result case 1. 1.4D
	PointSupportMX(support, wantMax)	Returns the maximum or minimum Moment X for the specified point support	PointSupportMX("PS1", true)	Returns the maximum Moment X for point support PS1 across all result cases
	PointSupportMY(result, support)	Returns Moment Y for the specified point support for the specified result case	PointSupportMY("1. 1.4D", "PS1")	Returns Moment Y for point support PS1 for result case 1. 1.4D
	PointSupportMY(support, wantMax)	Returns the maximum or minimum Moment Y for the specified point support	PointSupportMY("PS1", true)	Returns the maximum Moment Y for point support PS1 across all result cases
<b>Line Support Results</b> <a href="#">(BACK TO TOP)</a>	LineSupportForce(result, support)	Returns the force for the specified line support for the specified result case	LineSupportForce("1. 1.4D", "LS1")	Returns the force for line support LS1 for result case 1. 1.4D
	LineSupportForce(support, wantMax)	Returns the maximum or minimum force for the specified line support	LineSupportForce("LS1", true)	Returns the maximum force for line support LS1 across all result cases
	LineSupportMoment(result, support)	Returns the moment for the specified line support for the specified result case	LineSupportMoment("1. 1.4D", "LS1")	Returns the moment for line support LS1 for result case 1. 1.4D
	LineSupportMoment(support, wantMax)	Returns the maximum or minimum moment for the specified line support	LineSupportMoment("LS1", true)	Returns the maximum moment for line support LS1 across all result cases
<b>Design</b>	SlabReinforcingWeight()	Returns the weight of		

<b>Results</b> <a href="#">(BACK TO TOP)</a>	SlabGrossConcreteWeight()	the reinforcement Returns the gross concrete weight		
<b>Report</b> <a href="#">(BACK TO TOP)</a>	AddTable(title)	Adds a specified table to the report	AddTable("Plate Forces")	Adds the Plate Forces table to the report
	SetTableWidth(title, fraction)	Sets the designated table's page width to the specified fraction	SetTableWidth("Vertices", 0.5)	Sets the Vertices table's page width to half
	AddGraphicToReport(title)	Adds the current graphic window with a specified title to the report	AddGraphicToReport("Slab Design")	Adds the current graphic window with a title of "Slab Design" to the report
	ExportReport(path)	Exports the report to a specified path	ExportReport("C:/Users/your.login/Desktop/Report.pdf")	Saves the report as a .pdf on the desktop for the specified user

## 5.3 External Scripts

### Running External Scripts

1. Opening External Scripts
  - a. Type Browse into the command line to launch the Open dial box and select the script text file to use.
  - b. Directly type the path of the script text file into to the command line and press Enter.
2. Example Scripts
  - a. [Two-Way Slab Generator](#)
  - b. [Refine Mesh](#)
  - c. [Optimize Thickness](#)

## 5.4 Example: Two-Way Slab Generator

### Two-Way Slab Generator

```
//Generates a two-way floor slab
SetUnits("Kips & Feet");
DeleteAll();

//input
var xBays = 3;
var yBays = 2;
var xSpacing = 18;
var ySpacing = 12;
var thickness = 1.0;
var material = "Concrete (F'c = 5 ksi)";
var supportSize = 1.0;

//define locations
var locations = new List<Location>();
for(var i = 0; i <= yBays; i++)
{
    for(var j = 0; j <= xBays; j++)
    {
```

```

        var x = j * xSpacing;
        var y = i * ySpacing;
        locations.Add(new Location(x, y));
    }
}

//define the slab
SetMaterial(material);
var i = 0;
var j = xBays;
var k = (xBays + 1) * (yBays + 1) - 1;
var l = k - xBays;
var slab = AddBoundary(locations[i], locations[j], locations[k], locations[l]);
Thickness(slab, thickness);

//add supports
foreach(var location in locations)
{
    var support = AddPointSupport(location);
    SquarePointSupportSize(support, supportSize);
    FixPointSupport(support, "DZ");
    FixPointSupport(support, "RX");
    FixPointSupport(support, "RY");
    RefinePointSupport(support, supportSize, supportSize/10, supportSize/3);
}

```

---

## 5.5 Example: Refine Mesh

### Refine Mesh

```

//Use a loop to refine the mesh of the ACI350 tank wall example project

SetUnits("Kips & Inches");

//open the ACI350 tank wall example before running the script

//Try finer and finer mesh until the extreme Mx moment stops changing
var meshSetting = 100;
var delta = 100.0; //initialize to a big value
var moment = -100.0; //initialize to a big value
int i = 0; //keep track of the number of iterations needed to find the converged value
var tolerance = 1.0; //stop when the moment changes by less than 1 k-in/in
while(delta > tolerance)
{
    i++;
    meshSetting *= 2;
    SetMeshCustom(meshSetting);
    await Analysis();
    //interested in the negative moment at the side support
    var newMoment = MomentX(false);
    delta = Math.Abs(moment - newMoment);
    moment = newMoment;
}

//Print out the final results
$"Mux = {moment:f1}; Final Delta = {delta:f2}; Trials = {i}"

```

## 5.6 Example: Optimize Thickness

### Optimize Thickness

```
//Vary the thickness of a concrete slab to minimize the amount of reinforcement needed
//Non-linear problem since the larger thicknesses will increase the load on the
structure
//and add to minimum steel requirements

SetUnits("Kips & Inches");
DeleteAll();

//define the model
var width = 480.0;
var height = 48.0;
var thickness = 12.0;
var slab = AddRectangleBoundary(0, 0, width, height);
Thickness(slab, 12);
DoubleMat_Optimize(slab, true, 1.5, true, 1.5, ("#6", 12), ("#6", 6), ("#8", 6));
SetMaterial("Concrete (F'c = 5 ksi)");

var supportT = 16.0;
var leftSupport = AddLineSupport("left", -width/2, -height/2, -width/2, height/2,
supportT);
PinLineSupport(leftSupport);
var rightSupport = AddLineSupport("right", width/2, -height/2, width/2, height/2,
supportT);
PinLineSupport(rightSupport);

//add live load
RectangularLoad("L", 0, 0, height, height, -50.0/144/1000, 0, 0);

//Add one inch to the slab thickness, find the weight of reinforcement required, save
the most efficient thickness
var steelWt = 9999999999.9; //initialize to a big value
var t = thickness;
var bestThickness = t;
while(t <= 30.0) //stop at a 30" slab
{
    await Design();
    //has to pass shear and flexure checks to be considered
    if(Status("Slab Flexural Steel") == 2 && Status("Slab Shear") == 2)
    {
        var wt = SlabReinforcingWeight();
        if(wt < steelWt)
        {
            steelWt = wt;
            bestThickness = t;
        }
    }
    t += 1.0;
    Thickness(slab, t);
}
Thickness(slab, bestThickness);

//Print out the final results
$"Optimal Thickness = {bestThickness} inches; Reinforcing Wt = {steelWt:f4} kips"
```



